LilyPond

Le système de gravure musicale

Utilisation des programmes

L'équipe de développement de LilyPond

Ce document constitue le manuel d'utilisation des programmes de GNU LilyPond 2.19.31. De plus, ce manuel suggère des « bonnes pratiques » pour une utilisation plus efficace.

Pour connaître la place qu'occupe ce manuel dans la documentation, consultez la page Section "Manuels" dans *Informations générales*.

Si vous ne disposez pas de certains manuels, la documentation complète se trouve sur `http://www.lilypond.org/`.

Pour LilyPond version 2.19.31

Table des matières

1 Exécution de lilypond

Ce chapitre passe en revue ce qui se passe lorsque vous lancez LilyPond.

1.1 Utilisation habituelle

La plupart des utilisateurs de LilyPond le font au travers d'une interface graphique (*GUI* pour *graphical user interface*). Si vous ne l'avez pas encore parcouru, lisez le Section "Tutoriel" dans *Manuel d'initiation*. Si vous utilisez un éditeur alternatif pour rédiger vos fichiers LilyPond, référez-vous à la documentation de celui-ci.

1.2 Utilisation en ligne de commande

Nous nous intéresserons ici aux spécificités de LilyPond employé en ligne de commande. La ligne de commande permet de faire appel à certaines options particulières. D'autre part, certains utilitaires associés, tel que `midi2ly`, ne sont disponibles qu'en ligne de commande.

Par « ligne de commande », nous entendons l'interface de commande du système. Les utilisateurs de Windows seront certainement plus familiers des termes « fenêtre DOS » ou « invite de commande ». Quant aux utilisateurs de MacOS X, ils connaissent assurément les termes « console » et « terminal ». Les paramétrages spécifiques au système MacOS font l'objet d'un Section "chapitre particulier" dans *Informations générales*.

Notre propos n'est pas ici d'expliquer ce qu'est l'interface de commande pour un système informatique ni comment elle fonctionne. Aussi, si vous ne savez de quoi il retourne, nous vous renvoyons aux nombreuses documentations que vous pourrez trouver sur ce sujet.

Lancement de LilyPond

L'exécutable `lilypond` en ligne de commande se lance ainsi :

```
lilypond [option]... fichier...
```

Lorsque le fichier est fourni sans extension, LilyPond présume qu'il s'agit de `.ly`. Pour interpréter directement l'entrée standard (*stdin*), fournissez un tiret (`-`) en lieu et place de *fichier*.

Le traitement de `monfichier.ly` produira `monfichier.ps` et `monfichier.pdf`. Vous pouvez spécifier plusieurs fichiers à la fois ; ils seront traités indépendamment les uns des autres.[1]

Lorsque `monfichier.ly` contient plus d'une section `\book`, les fichiers produits – à partir du deuxième – seront numérotés. Par ailleurs, la valeur affectée à `output-suffix` sera insérée entre la racine et le numéro. Par exemple, un fichier *racine* qui contiendrait

```
#(define output-suffix "violon")
\score { ... }
#(define output-suffix "cello")
\score { ... }
```

fournira grâce à LilyPond `racine-violon.pdf` et `racine-cello-1.pdf`.

Utilisation de LilyPond avec les fonctionnalités standard de l'interpréteur

Dans la mesure où LilyPond est une application qui fonctionne en ligne de commande, les fonctionnalités de l'interpréteur utilisé pour lancer LilyPond peuvent se révéler utiles.

Par exemple,

[1] Le statut de GUILE n'étant pas réinitialisé après traitement d'un fichier `.ly`, veillez à ne pas modifier les réglages par défaut du système à partir d'assertions en Scheme.

```
lilypond *.ly
```

traitera tous les fichiers LilyPond présents dans le répertoire en cours.

Rediriger, par exemple dans un fichier, ce qui est émis à l'écran peut s'avérer utile.

```
lilypond fichier.ly 1> stdout.log

lilypond fichier.ly 2> stderr.log

lilypond fichier.ly &> tous.log
```

redirigeront respectivement le « verbiage normal », les erreurs ou tout, dans un fichier texte.

Consultez avant tout la documentation de votre interpréteur habituel – terminal, console, etc. – pour vérifier qu'il prend en charge les options dans cette syntaxe.

Voici comment traiter un jeu de fichiers répartis dans un répertoire donné ainsi que tous ses différents sous-répertoires. Les fichiers résultants seront regroupés dans le répertoire à partir duquel la commande a été exécutée, non selon l'emplacement des fichiers sources.

```
find . -name '*.ly' -exec lilypond '{}' \;
```

Cette commande, bien qu'effective uniquement dans un terminal, devrait être fonctionnelle aussi pour les utilisateurs de MacOS X.

Les utilisateurs de windows utiliseront l'instruction

```
forfiles /s /M *.ly /c "cmd /c lilypond @file"
```

dans l'`interpréteur de commandes`, qui se trouve normalement sous `Démarrer > Accessoires > Interpréteur de commandes` ou, pour la version 8, en faisant une recherche sur « interpréteur de commande ».

Par ailleurs, il est possible de spécifier de manière explicite le chemin d'accès au dossier comportant des sous-répertoires où se trouvent les fichiers sources, à l'aide de l'option `/p` :

```
forfiles /s /p C:\Documents\MesPartitions /M *.ly /c "cmd /c lilypond @file"
```

Dans le cas où ce chemin d'accès comporte des espaces, l'intégralité de ce chemin devra être borné par des guillemets informatiques :

```
forfiles /s /p "C:\Documents\Mes Partitions" /M *.ly /c "cmd /c lilypond @file"
```

Options basiques de `lilypond`

Différentes options sont disponibles en ligne de commande :

`-b, --bigpdfs`

Les fichiers PDF générés peuvent être beaucoup plus lourd que la normale selon le degré d'optimisation des fontes. Néanmoins, lorsque plusieurs fichiers PDF sont inclus dans un document `pdftex`, `xetex` ou `luatex`, ils peuvent faire l'objet d'un traitement supplémentaire par `ghostscript` afin de fusionner les données de fontes redondantes et ainsi obtenir un fichier PDF *significativement* plus léger.

```
lilypond -b monfichier
```

Puis lancer `ghostscript` :

```
gs -q -sDEVICE=pdfwrite -o gsout.pdf monfichier.pdf
```

`pdfsizeopt.py` vient alors en complément pour optimiser encore la taille du fichier :

```
pdfsizeopt.py --use-multivalent=no gsout.pdf final.pdf
```

`-d, --define-default=`*`variable`*`=`*`valeur`*

Voir [Options avancées de lilypond], page 5.

`-e, --evaluate=`*expr*

Évalue l'expression Scheme *expr* avant d'analyser tout fichier `.ly`. Lorsque vous spécifiez l'option `-e` à plusieurs reprises, l'évaluation sera faite en séquence.

Dans la mesure où l'expression est évaluée par le module `guile-user`, vous devrez, dès lors que *expr* utilise des définitions, spécifier

```
lilypond -e '(define-public a 42)'
```

en ligne de commande, et ajouter la ligne

```
#(use-modules (guile-user))
```

en tête de votre fichier `.ly`.

> **Note :** Les utilisateurs de Windows doivent utiliser des guillemets doubles `"` en lieu et place des guillemets simples `'`.

`-f, --format=`*format*

Détermine le format à produire. Il peut s'agir de `ps`, `pdf` ou `png`.

Exemple : `lilypond -fpng` *monfichier*`.ly`

`-h, --help`

Affiche un résumé des commandes.

`-H, --header=`*CHAMP*

Recopie le champ d'entête dans le fichier `RACINE.`*CHAMP*.

`-i, --init=`*fichier*

Définit *fichier* (par défaut `init.ly`) en tant que fichier d'initialisation.

`-I, --include=`*répertoire*

Ajoute *répertoire* au chemin de recherche pour les inclusions.

Vous pouvez mentionner plusieurs fois l'option `-I`, auquel cas la recherche commencera dans le premier répertoire inclus et, si le fichier en question ne s'y trouve pas, les répertoires suivants seront examinés l'un après l'autre.

`-j, --jail=`*user*`,`*group*`,`*jail*`,`*dir*

Lance `lilypond` dans un environnement protégé.

L'option `--jail` est une alternative qui offre plus de flexibilité que l'option `--safe` lorsque LilyPond est installé sur un serveur web ou traite des fichiers externes – voir [Options avancées de lilypond], page 5.

L'option `--jail` va détourner la racine de `lilypond` sur *jail* juste avant d'effectuer la compilation à proprement parler. L'utilisateur et le groupe sont modifiés en conséquence, et le répertoire en cours devient *dir*. Ces réglages assurent – du moins en théorie – l'impossibilité de s'échapper de la cellule. Notez cependant que, pour que l'option `--jail` soit fonctionnelle, `lilypond` doit être lancé en tant qu'administrateur – ce qui se réalise aisément à l'aide de la commande `sudo`.

La création d'un environnement sécurisé requiert quelques précautions dans la mesure où LilyPond doit disposer de tout ce dont il a besoin pour compiler le fichier source **à l'intérieur de la cellule**. L'ermitage, avant d'être viable, requiert donc les étapes suivantes :

Création d'un système de fichiers indépendant

L'intérêt d'un système de fichiers dédié à LilyPond réside dans le fait qu'on peut le brider à l'aide des options `noexec`, `nodev` et `nosuid`. Il sera de fait impossible de lancer des exécutables ou d'écrire sur un

périphérique à partir de LilyPond. Si vous n'avez pas l'intention de créer un tel système sur une partition séparée, vous pouvez avoir recours à un pseudo-périphérique (*loop device*) monté à partir d'un simple fichier de taille raisonnable. D'autre part, le recours à un système de fichiers indépendant permet de limiter l'espace dévolu à LilyPond.

Création d'un utilisateur spécifique

L'utilisation de LilyPond au sein de la cellule devrait être réservé à un utilisateur aux droits restreints. Il faudra donc créer un utilisateur et un groupe spécifiques – disons `lily/lily` – qui n'aura accès en écriture qu'à un unique répertoire déterminé par la valeur de *dir*.

Agencement des lieux

LilyPond a besoin d'un certain nombre de fichiers pour pouvoir tourner correctement. Ces fichiers devront donc tous se retrouver dans l'environnement protégé, distribués selon la même arborescence que dans le système d'origine. Ainsi l'intégralité de l'installation de LilyPond (en principe `/usr/share/lilypond`) doit y être dupliquée.

En cas de problème, lancer LilyPond en utilisant `strace` devrait vous permettre de déterminer quels fichiers manquent à l'appel.

Lancement de LilyPond

Dans un environnement protégé monté avec l'option `noexec`, il est impossible de lancer un quelconque programme extérieur. LilyPond ne saurait donc avoir recours à un moteur de traitement qui le mettrait dans cette situation. Comme nous l'avons vu plus haut, LilyPond sera lancé avec les privilèges de l'administrateur – privilèges qu'il perdra aussitôt –, ce qui peut nécessiter le recours à la commande `sudo`. Il est par ailleurs judicieux de limiter le temps processeur alloué à LilyPond – grâce à `ulimit -t` par exemple – ainsi que, si votre système le permet, la taille de la mémoire. Voir aussi [Exécution de LilyPond en mode protégé], page 11.

`-l, --loglevel=`*DEGRÉ*

Règle le niveau de verbosité des messages console à *DEGRÉ*. Les différentes valeurs sont :

`NONE` Aucun verbiage, même pas les messages d'erreur.

`ERROR` Uniquement les messages d'erreur ; pas de message d'avertissement ni de progression.

`WARN` Messages d'avertissement ou d'erreur ; pas d'information de progression.

`BASIC_PROGRESS`

Information de progression basique (réussite) et avertissements ou erreurs.

`PROGRESS` Toutes les informations de progression, avertissements et erreurs.

`INFO (par défaut)`

Informations de progression, avertissements et erreurs, ainsi que d'autres informations relatives à l'exécution.

`DEBUG` Tout ce qui peut être affiché, y compris le verbiage utile au débogage.

`-o, --output=`*FICHIER* ou *RÉPERTOIRE*

Détermine le nom par défaut du fichier résultant à *FICHIER* ; lorsque l'argument *RÉPERTOIRE* correspond à un répertoire déjà existant, c'est là que les fichiers

résultants seront déposés. Le suffixe adéquat sera ajouté (par ex. `.pdf` pour du pdf) dans tous les cas.

`--ps` Génère du PostScript.

`--png` Génère une image par page, au format PNG ; ceci sous-entend l'utilisation de `--ps`. La résolution de l'image, en DPI, peut se régler en ajoutant par exemple

```
-dresolution=110
```

`--pdf` Génère du PDF. Ceci sous-entend l'utilisation de `--ps`.

`-v, --version`
Affiche le numéro de version.

`-V, --verbose`
Active le mode verbeux : affichage de l'intégralité du chemin d'accès de chaque fichier, et information des temps de traitement.

`-w, --warranty`
Affiche les informations de garantie applicables à GNU LilyPond – il est livré **SANS GARANTIE** !

Options avancées de `lilypond`

`-d`*`[nom-option]`*`=`*`[valeur]`*`, --define-default=`*`[nom-option]`*`=`*`[valeur]`*
Affecte la valeur Scheme *valeur* à l'option interne *nom-option* du programme.

```
-dbackend=svg
```

En l'absence de *valeur*, le programme utilisera *#t*. Préfixer *nom-option* d'un `no-` vous permet de désactiver une option. Ainsi,

```
-dno-point-and-click
```

revient au même que

```
-dpoint-and-click=#f
```

Voici les différentes options disponibles, ainsi que leur valeur par défaut.

Symbole	**Valeur**	**Observations**
`anti-alias-factor`	`1`	Adopte une résolution supérieure (selon le facteur donné),puis réduit au niveau du résultat afin d'éviter les « distorsions » des images `PNG`.
`aux-files`	`#t`	Génère les fichiers `.tex`, `.texi` et `.count` pour le moteur de rendu `EPS`.
`backend`	`ps`	Détermine le format de sortie à utiliser par le moteur de traitement. Les fichiers PostScript (format par défaut) incluent les fontes `TTF`, `Type1` et `OTF`, et aucune substitution ne sera opérée pour ces fontes. Si vous utilisez des jeux de caractères orientaux, le fichier aura vite fait d'atteindre une taille conséquente.

`eps` Génère du PostScript encapsulé. Chaque page (système) fera l'objet d'un fichier `EPS` particulier, sans fontes, auquel sera associé un fichier `EPS` qui, lui, contiendra toutes les pages (systèmes) et les fontes. Notez qu'il s'agit du mode que `lilypond-book` utilise par défaut.

`null` Ne génère aucun fichier imprimable. Cette option est équivalente à `-dno-print-pages`.

`scm` Recopie littéralement les commandes Scheme internes de formatage.

`svg` Génère du *Scalable Vector Graphics*. Cette option permet de créer un fichier SVG par page. Dans la mesure où aucune information concernant les fontes n'est incorporée, elles devront être installées afin d'obtenir le meilleur rendu possible. Il est préférable de ne pas recourir aux « alias de police » ni aux listes de fontes si le visionneur de fichier SVG ne peut le traîter correctement. L'option `svg-woff` – voir ci-après – permet d'utiliser les fontes woff avec le moteur SVG.

Note à propos du moteur SVG : Les polices par défaut de LilyPond – `LilyPond Serif`, `LilyPond Sans Serif` et `LilyPond Monospace` – ne sont que des alias au niveau local. Le recours au moteur SVG requiert donc que ces polices par défaut soient explicitement définies :

```
\paper  {
  #(define fonts
    (make-pango-font-tree "TeX Gyre Schola"
                          "TeX Gyre Heros"
                          "TeX Gyre Cursor"
                          (/ staff-height pt 20)))
}
```

Voir aussi Section "Choix des fontes par défaut" dans *Manuel de notation*.

`check-internal-types` `#f` Vérifie qu'à chaque propriété est bien affecté un type.

`clip-systems` `#f` Extrait des fragments musicaux d'une partition. Ceci requiert que la fonction `clip-regions` a été définie au sein du bloc `\layout` – voir Section "Extraction de fragments musicaux" dans *Manuel de notation*. Bien entendu, aucun fragment ne sera extrait si l'on utilise l'option `-dno-print-pages`

`datadir` Détermine le préfixe des fichiers de données (lecture seule).

debug-gc	#f	Génère une copie brute de la mémoire, aux fins de débogage.
debug-gc-assert-parsed-dead	#f	Pour débogage de la mémoire : s'assure que toute référence à des objets analysés est effacée. Il s'agit d'une option interne qui est automatiquement activée par l'option `-ddebug-gc'.
debug-lexer	#f	Débogage de l'analyseur lexical *flex*.
debug-page-breaking-scoring	#f	Purge les calculs des configurations de saut de page.
debug-parser	#f	Débogage de l'analyseur *bison*.
debug-property-callbacks	#f	Débogage des chaînes de *callback* cycliques.
debug-skylines	#f	Débogage des lignes d'horizon.
delete-intermediate-files	#t	Supprime les fichiers .ps inutiles crées lors de la compilation.
dump-cpu-profile	#f	Génère une copie brute des informations de timing (dépend du système).
dump-profile	#f	Génère une copie brute de la mémoire et des temps de traitement pour chaque fichier.
dump-signatures	#f	Génère une copie des signatures de chaque système. Cette option est utilisée pour les tests de régression.
eps-box-padding	#f	Décale le bord gauche du typon EPS d'une valeur donnée en millimètres.
gs-load-fonts	#f	Charge les fontes grâce à Ghostscript.
gs-load-lily-fonts	#f	Limite les fontes chargées par Ghostscript aux seules fontes LilyPond.
gui	#f	Travaille silencieusement, et redirige tout le verbiage dans un fichier journal.

Note à l'attention des utilisateurs de Windows : toutes les informations concernant le traitement apparaissent au fur et à mesure dans l'interpréteur de commandes lorsque vous lancez le programme lilypond.exe, à l'inverse de lilypond-windows.exe qui vous renvoie simplement la main. L'option -dgui vous permettra alors de rediriger ces informations dans un fichier journal.

help	#f	Affiche cette aide.
include-book-title-preview	#t	Inclut les titres de l'ouvrage dans les images de prévisualisation.

include-eps-fonts	#t	Inclut les fontes dans chaque fichier EPS contenant un système.
include-settings	#f	Inclut un fichier contenant les réglages globaux. Ce fichier sera inclus avant traitement de la partition.
job-count	#f	Traite plusieurs fichiers en parallèle, selon le nombre de *jobs*.
log-file	#f [fichier]	Fournir TOTO en second argument redirigera la sortie dans le fichier journal TOTO.log.
max-markup-depth	1024	Profondeur maximale de l'arborescence de *markups*. Si un *markup* était plus profond, part du principe qu'on aboutira pas, émet un avertissement et renvoie alors un *markup* vide.
midi-extension	"midi"	Détermine l'extension par défaut des fichiers MIDI, selon la chaîne donnée en argument.
music-strings-to-paths	#f	Convertit les chaînes textuelles en chemins lorsque les glyphes font partie d'une fonte musicale.
paper-size	\"a4\"	Détermine la taille par défaut du papier. Veillez à ne pas oublier d'encadrer la valeur par des guillemets échappés (\").
pixmap-format	png16m	Détermine le format de sortie en images pixélisées pour Ghostscript.
point-and-click	#t	Ajoute les liens « point & click » à la sortie PDF ou SVG. Voir Section 4.1 [Pointer-cliquer], page 42.
preview	#f	Génère une prévisualisation en plus de la sortie normale.

Cette option, disponible dans tous les formats de sortie imprimables – `pdf`, `png`, `ps`, `eps` et `svg` – génère un fichier de la forme `monFichier.preview.extension` comprenant le titrage et le premier système. S'il existe plusieurs sections `\book`, `\bookpart`, ce fichier contiendra les titrage et premier système de chacun des `\book`, `\bookpart` et `\score`, dès lors que la variable `print-all-headers` du bloc `\paper` est activée.

Pour l'éviter, utilisez conjointement l'une des options `-dprint-pages` ou `-dno-print-pages` selon vos besoins.

print-pages	#t	Génère l'intégralité des pages de la partition. L'option `-dno-print-pages` est particulièrement utile lorsqu'utilisée conjointement avec l'option `-dpreview`.

`profile-property-accesses`	`#f`	Enregistre des statistiques des appels à la fonction `get_property()`.
`protected-scheme-parsing`	`#t`	Continue en dépit des erreurs que l'analyseur syntaxique détecterait dans du code Scheme inclus. Lorsque basculé sur `#f`, stoppe le traitement s'il y a erreur et affiche une trace de la pile.
`read-file-list`	`#f [fichier]`	Spécifie un fichier listant les différents fichiers sources à traiter.
`relative-includes`	`#f`	Face à une instruction `\include`, recherche les fichiers à inclure relativement à l'endroit où se trouve le fichier en cours de traitement plutôt que par rapport au fichier maître.
`resolution`	`101`	Détermine, en `dpi`, la résolution des pixmaps `PNG` à générer selon la valeur donnée.
`safe`	`#f`	Ne pas avoir une confiance aveugle dans le code `.ly`.

Lorsque LilyPond est accessible au travers d'un serveur web, il est **impératif** d'utiliser les options `--safe` ou `--jail`. L'option `--safe` aura pour effet d'empêcher tout code Scheme inclus de mettre en péril votre installation grâce à quelque chose du style

```
#(s ystem "rm -rf /") % trop dangeureux à écrire sans faute
{
  c4^$(ly:gulp-file "/etc/passwd") % malveillant mais pas destructeur
}
```

L'option `-dsafe` forcera l'évaluation, au fil de l'eau et par un module sécurisé, des expressions Scheme contenues dans le fichier source. Ce module sécuritaire, dérivé du module GUILE `safe-r5rs`, ajoute un certain nombre de fonctions – listées dans `scm/safe-lily.scm` – à l'API de LilyPond.

De plus, le mode *safe* ne permet ni l'utilisation de directives `\include` ni le recours aux obliques inversées (*backslash*) dans les chaînes TeX. L'import de variables LilyPond dans du code Scheme n'est pas possible en mode sécurisé.

L'option `-dsafe` **ne détecte pas** l'utilisation abusive des ressources. Il est donc possible que le programme finisse par rester sans réponse si on lui envoie une boucle sans fin. C'est la raison pour laquelle nous recommandons, lorsque LilyPond tourne sur un serveur accessible au public, d'en limiter aussi bien les ressources processeur que mémoire.

Notez bien que l'utilisation du mode sécuritaire empêchera aussi la compilation d'un certain nombre de fragments LilyPond. L'option `--jail` est dans ce cas une excellente alternative en terme de sécurité, même si elle requiert plus de temps à mettre en place – voir [Options basiques de lilypond], page 2.

`separate-log-files`	`#f`	Pour les fichiers `FICHIER1.ly`, `FICHIER2.ly`, etc. enregistre le déroulement dans les journaux `FICHIER1.log`, `FICHIER2.log`...
`show-available-fonts`	`#f`	Liste le nom des fontes disponibles.

strict-infinity-checking	#f	Force le crash en présence des points d'exception de virgule flottante Inf ou NaN – infini ou non-nombre.
strip-output-dir	#t	Supprime, lors du nommage des fichiers résultant, la partie correspondant au répertoire des fichiers sources.
strokeadjust	#f	Force l'ajustement des traits PostScript. Cette option trouve toute son utilité pour générer du PDF à partir de PostScript – l'ajustement des traits est en principe automatiquement activé pour les périphériques bitmap à faible résolution. Sans cette option, les visionneurs de PDF ont tendance à ne pas rendre de manière constante l'épaisseur des hampes dans les résolutions habituelles des écrans. Bien que n'affectant pas notoirement la qualité d'impression, cette option accroit notablement la taille des fichiers PDF.
svg-woff	#f	Utilise, avec le moteur SVG, les fontes woff. Lorsque celles-ci sont utilisées correctement, nul n'est besoin d'installer les fontes que les fichiers SVG utiliseront dans l'environnement du visionneur. Néanmoins, LilyPond ne dispose pas de fichier de fonte woff textuelle. La présence du fichier de fonte woff est un prérequis.
trace-memory-frequency	#f	Enregistre l'utilisation de la cellule Scheme plusieurs fois par seconde, dans les fichiers FICHIER.stacks et FICHIER.graph.
trace-scheme-coverage	#f	Enregistre la couverture des fichiers Scheme dans FILE.cov.
verbose	#f	Passe en mode verbeux, ce qui correspond à un niveau de journalisation DEBUG (lecture seule).
warning-as-error	#f	Considère tous les messages d'avertissement et « erreur de programmation » comme étant de véritables erreurs.

Variables d'environnement

lilypond reconnaît les variables d'environnement suivantes :

LILYPOND_DATADIR
: Cette variable spécifie le répertoire où seront recherchés par défaut les différentes versions des messages ainsi qu'un certain nombre de fichiers nécessaires au traitement. Il devrait contenir les sous-répertoires ly/, ps/, tex/, etc.

LANG
: Cette variable détermine la langue dans laquelle seront émis les messages.

`LILYPOND_LOGLEVEL`

Cette variable détermine le niveau par défaut de verbosité. En l'absence de niveau explicite – autrement dit la ligne de commande ne comporte pas de `--loglevel` – c'est cette valeur qui sera utilisée.

`LILYPOND_GC_YIELD`

Cette variable permet d'ajuster l'empreinte mémoire et le rendement de la machine. Il s'agit en fait d'un pourcentage d'allocation de mémoire : lorsqu'il est élevé, le programme favorisera l'utilisation de la mémoire ; une faible valeur consommera plus de temps processeur. Par défaut, cette valeur est fixée à `70`.

Exécution de LilyPond en mode protégé

Paramétrer un serveur afin qu'il puisse faire fonctionner LilyPond en mode protégé sur un pseudo-périphérique est une tâche sensible. Les différentes étapes à suivre sont répertoriées ci-dessous. Les exemples qu'elle comportent proviennent d'une distribution GNU/Linux Ubuntu et nécessiteront l'utilisation de `sudo` autant que de besoin.

- Installation des paquetages nécessaires : LilyPond, Ghostscript et ImageMagick.
- Création de l'utilisateur `lily` :

  ```
  adduser lily
  ```

  Ceci, par la même occasion, créera un groupe spécifique pour l'utilisateur `lily` ainsi que son répertoire personnel `/home/lily`.
- Création, dans le répertoire personnel de l'utilisateur `lily`, d'un espace agissant en tant que système de fichiers :

  ```
  dd if=/dev/zero of=/home/lily/loopfile bs=1k count= 200000
  ```

  Cette commande a créé un fichier de 200 MB utilisable par le « système protégé ».
- Création d'un pseudo-périphérique, génération d'un système de fichiers et chargement de celui-ci, puis création d'un répertoire accessible en écriture pour l'utilisateur `lily` :

  ```
  mkdir /mnt/lilyloop
  losetup /dev/loop0 /home/lily/loopfile
  mkfs -t ext3 /dev/loop0 200000
  mount -t ext3 /dev/loop0 /mnt/lilyloop
  mkdir /mnt/lilyloop/lilyhome
  chown lily /mnt/lilyloop/lilyhome
  ```
- Affectation, au niveau configuration du serveur, de `/mnt/lilyloop` en tant que JAIL et `/lilyhome` en tant que DIR.
- Création d'une arborescence, dans l'espace protégé, et recopie de tous les fichiers nécessaires – voir le script plus loin.

  Le recours à l'utilitaire `sed` permet de créer les commandes de copie de tout ce qui est nécessaire à un exécutable :

  ```
  for i in "/usr/local/lilypond/usr/bin/lilypond" "/bin/sh" "/usr/bin/;  \
    do ldd $i | sed 's/.*=> \/\(.*\/\)\([^(]*\).*/mkdir -p \1 \&\&  \
      cp -L \/\1\2 \1\2/' | sed 's/\t\/\(.*\/\)\(.*\) (.*)$/mkdir -p  \
        \1 \&\& cp -L \/\1\2 \1\2/' | sed '/.*=>.*/d'; done
  ```

Exemple de script fonctionnel en 32-bit sur Ubuntu 8.04

```
#!/bin/sh
## les réglages par défaut

username=lily
```

```
home=/home
loopdevice=/dev/loop0
jaildir=/mnt/lilyloop
# le préfixe (sans slash au début !)
lilyprefix=usr/local
# le répertoire du système où lilypond est installé
lilydir=/$lilyprefix/lilypond/

userhome=$home/$username
loopfile=$userhome/loopfile
adduser $username
dd if=/dev/zero of=$loopfile bs=1k count=200000
mkdir $jaildir
losetup $loopdevice $loopfile
mkfs -t ext3 $loopdevice 200000
mount -t ext3 $loopdevice $jaildir
mkdir $jaildir/lilyhome
chown $username $jaildir/lilyhome
cd $jaildir

mkdir -p bin usr/bin usr/share usr/lib usr/share/fonts $lilyprefix tmp
chmod a+w tmp

cp -r -L $lilydir $lilyprefix
cp -L /bin/sh /bin/rm bin
cp -L /usr/bin/convert /usr/bin/gs usr/bin
cp -L /usr/share/fonts/truetype usr/share/fonts

# la formule magique de recopie des bibliothèques
for i in "$lilydir/usr/bin/lilypond" "$lilydir/usr/bin/guile" "/bin/sh"  \
  "/bin/rm" "/usr/bin/gs" "/usr/bin/convert"; do ldd $i | sed 's/.*=>  \
    \/\(.*\/\)\([^(]*\).*/mkdir -p \1 \&\& cp -L \/\1\2 \1\2/' | sed  \
      's/\t\/\(.*\/\)\(.*\) (.*)$/mkdir -p \1 \&\& cp -L \/\1\2 \1\2/'  \
        | sed '/.*=>.*/d'; done | sh -s

# les fichiers partagés pour ghostscript...
      cp -L -r /usr/share/ghostscript usr/share
# les fichiers partagés pour ImageMagick
      cp -L -r /usr/lib/ImageMagick* usr/lib

### Partant du principe que test.ly est dans /mnt/lilyloop/lilyhome,
### on devrait pouvoir lancer :
### Attention : /$lilyprefix/bin/lilypond est un script qui
### définit LD_LIBRARY_PATH - c'est primordial
      /$lilyprefix/bin/lilypond -jlily,lily,/mnt/lilyloop,/lilyhome test.ly
```

1.3 Messages d'erreur

Différents messages d'erreur sont susceptibles d'apparaître au cours de la compilation d'un fichier :

Warning – Avertissement
: Ce type de message est émis lorsque LilyPond détecte quelque chose de suspect. Si vous avez demandé quelque chose qui sort de l'ordinaire, vous saurez probablement ce à quoi il est fait référence et ignorerez de tels messages sans remord. Néanmoins, les messages d'avertissement indiquent la plupart du temps une incohérence dans le fichier source.

Error – Erreur
: LilyPond a détecté une erreur. L'étape en cours, qu'il s'agisse de l'analyse, de l'interprétation des données ou bien du formatage, sera menée à son terme, puis LilyPond s'arrêtera.

Fatal error – Erreur fatale
: LilyPond est confronté à une anomalie bloquante. Ceci ne se produit que très rarement, et la plupart du temps en raison d'une installation défectueuse des fontes.

Scheme error – Erreur Scheme
: Les erreurs qui interviennent lors de l'exécution de code Scheme sont gérées par l'interpréteur Scheme. L'utilisation du mode verbeux (options `-V` ou `--verbose`) vous permettra de localiser l'appel de fonction délictueux.

Programming error – Erreur de programmation
: LilyPond est confronté à une incohérence interne. Ce type de message est destiné à venir en aide aux développeurs et débogueurs. En règle générale, vous pouvez tout simplement les ignorer. Parfois, il y en a tant qu'ils masquent ce qui pourrait vous intéresser. . .

Aborted (core dumped) – Abandon
: Ce type de message indique que LilyPond a planté en raison d'une grave erreur de programmation. La survenance d'un tel message est considérée comme de la plus haute importance. Si vous y étiez confronté, transmettez un rapport de bogue.

Lorsque l'avertissement ou l'erreur est directement lié au fichier source, le message est libellé sous la forme

```
fichier:ligne:colonne: message
contenu de la ligne litigieuse
```

Un saut de ligne est placé dans la ligne de code, indiquant l'endroit précis du problème, comme ici :

```
test.ly:2:19: erreur: n'est pas une durée: 5
  { c'4 e'
          5 g' }
```

Notez que ces coordonnées constituent l'approximation au mieux par LilyPond dans le code ayant déclenché l'avertissement ou l'erreur. En règle générale, erreurs et avertissements surviennent lorsque LilyPond rencontre quelque chose d'inattendu. Lorsque la ligne indiquée ne vous semble pas comporter d'élément litigieux, remontez de quelques lignes dans votre code.

Par ailleurs, des diagnostics peuvent être déclenchés à n'importe quel moment au cours des différentes étapes du traitement. Par exemple, lorsque certaines parties de la source sont traitées plusieurs fois – sortie MIDI et sortie imprimable – ou qu'une même variable musicale est utilisée dans plusieurs contextes, peut apparaître le même message à plusieurs reprises. Les diagnostics effectués à une étape avancée du traitement, tels que les contrôles de mesure, sont aussi susceptibles d'apparaître plusieurs fois.

Vous trouverez d'autres informations sur les erreurs au chapitre Section 1.4 [Quelques erreurs des plus courantes], page 14.

1.4 Quelques erreurs des plus courantes

Les conditions amenant aux erreurs qui suivent sont fréquentes, bien qu'elles ne soient pas évidentes ni facilement localisables. Nous espérons que ces explications vous aideront à les résoudre plus facilement.

La musique déborde de la page

Lorsque la musique s'épanche au delà de la marge droite ou bien semble anormalement comprimée, la raison en est le plus souvent une note à la durée erronée ; cela finit par provoquer le débordement de la dernière note d'une mesure. Rien ne s'oppose à ce que la dernière note d'une mesure ne s'arrête avant la barre de mesure ; on considère simplement qu'elle se prolonge sur la mesure suivante. Des débordements à répétition finissent par générer une musique comprimée ou qui sort de la page, pour la simple et bonne raison que les sauts de ligne automatiques ne peuvent intervenir qu'à la fin d'une mesure complète, autrement dit lorsque toutes les notes sont terminées avant la fin de la mesure.

> **Note :** Une durée erronée peut empêcher les sauts de ligne, ce qui conduit à une musique compressée, voire à un débordement de la page.

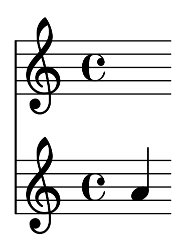
Une erreur de durée sera bien plus facilement localisable si vous positionnez régulièrement des contrôles de barre de mesure – voir Section "Vérification des limites et numéros de mesure" dans *Manuel de notation*.

Si vous tenez absolument à enchaîner de tels débordements, vous devrez insérer des barres de mesure invisibles là où vous souhaitez positionner un saut de ligne. Consultez le chapitre Section "Barres de mesure" dans *Manuel de notation* pour plus de détails.

Apparition d'une portée supplémentaire

Lorsque les contextes ne sont pas créés explicitement par la commande `\new`, ils le seront si la commande à exécuter n'est pas censée s'appliquer au contexte en cours. Pour des partitions simples, le fait que les contextes soient automatiquement créés rend bien des services, et c'est d'ailleurs le cas pour la majorité des exemples contenus dans les manuels de LilyPond. Cependant, la création implicite d'un contexte peut aboutir à l'apparition d'une portée « parasite ». On s'attend par exemple, en lisant le code qui suit, à ce que toutes les têtes de note soient en rouge, alors que le résultat nous présente deux portées et que les notes, placées sur la portée inférieure, restent en noir.

```
\override Staff.NoteHead.color = #red
\new Staff { a' }
```



Étant donné qu'aucun contexte `Staff` n'existe lorsque la dérogation est introduite, LilyPond le crée implicitement pour lui appliquer la directive considérée. Survient alors la commande `\new Staff` qui, à son tour, crée une nouvelle portée pour contenir les notes qui suivent. Voici la syntaxe correcte pour obtenir ces notes en rouge :

```
\new Staff {
  \override Staff.NoteHead.color = #red
```

```
  a'
}
```

Message d'erreur `Unbound variable %`

Ce message d'erreur, qu'il apparaisse sur le terminal ou en fin de fichier journal, est associé à un message du type « GUILE a signalé une erreur... ». Il survient à chaque fois qu'un commentaire *LilyPond* est indûment placé dans une routine *Scheme*.

Un commentaire LilyPond est introduit par le signe pourcent (`%`) et ne doit en aucun cas se trouver dans une routine Scheme. En Scheme, les commentaires s'introduisent par un point-virgule (`;`).

Message d'erreur `FT_Get_Glyph_Name`

Ce message d'erreur, qu'il apparaisse sur le terminal ou en fin de fichier journal, survient lorsqu'un fichier source contient des caractères non ASCII et qu'il n'a pas été enregistré avec un encodage UTF-8. Pour plus de détails, reportez-vous au chapitre Section "Caractères spéciaux" dans *Manuel de notation*.

staff-affinities devraient aller en ordre décroissant

Cet avertissement est émis lorsque la partition ne comporte pas de portée, comme par exemple une feuille de chant avec un contexte `ChordName` et un contexte `Lyrics`. Ce message disparaîtra dès lors que vous autoriserez l'un de ces contextes à se comporter comme une portée, à l'aide de l'instruction

```
\override VerticalAxisGroup.staff-affinity = ##f
```

que vous insérerez dès sa création. Pour plus d'information, reportez-vous à la rubrique Section "Espacement des lignes rattachées à des portées" dans *Manuel de notation*.

Message d'erreur unexpected `\new`

Un bloc `\score` ne peut contenir qu'**une seule** expression musicale. Si, par contre, il comporte plusieurs `\new Staff`, `\new StaffGroup` ou autres contextes introduits par une commande `\new` qui ne seraient pas bornés par des accolades `{ ... }` ou des doubles chevrons `<< ... >>` comme ici :

```
\score {
  % Invalide ! Génère l'erreur : syntax error, unexpected \new
  % en français : erreur de syntaxe : \new inattendu
  \new Staff { ... }
  \new Staff { ... }
}
```

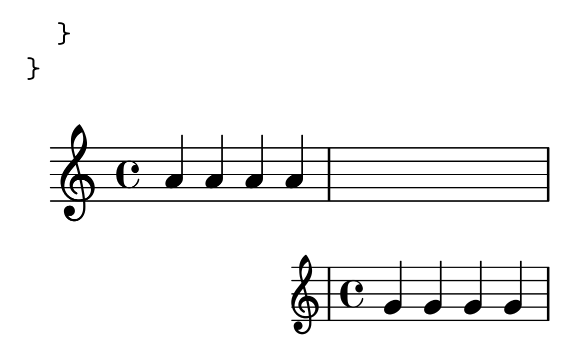
vous obtiendrez ce message d'erreur.

Cette erreur sera évitée dès lors que toutes les instances de `\new` sont bornées par des accolades ou des doubles chevrons.

Des accolades placeront ces clauses `\new` en séquence :

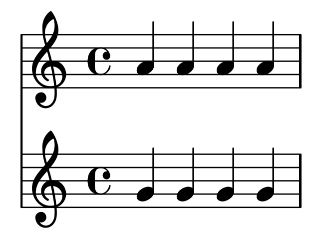
```
\score {
  {
    \new Staff { a' a' a' a' }
    \new Staff { g' g' g' g' }
```

```
  }
}
```



alors que des doubles chevrons les placeront en parallèle ; autrement dit, LilyPond les traitera simultanément :

```
\score {
  <<
    \new Staff { a' a' a' a' }
    \new Staff { g' g' g' g' }
  >>
}
```



Cette voix requiert un \voiceXx ou un réglage \shiftXx

Lorsque des notes affectées à des voix différentes et ayant la même orientation de hampe interviennent au même instant musical et qu'aucun décalage spécifique à la voix n'a été spécifié, LilyPond émet `Avertissement : Cette voix requiert un voiceXx ou un réglage shiftXx` (*warning: this voice needs a \voiceXx or \shiftXx setting*). Cet avertissement est émis même lorsque ces notes n'ont pas de hampe visible, comme par exemple des rondes, si les hampes des durées inférieures à ces même hauteurs avaient la même orientation.

N'oublions pas que l'orientation des hampes dépend de la position des notes sur la portée à moins que cette orientation n'ait été spécifiée, par exemple à l'aide d'un `\voiceOne` ou autre clause. En pareil cas, l'avertissement ne sera émis que lorsque les hampes auront la même orientation, autrement dit lorsque les notes seront dans la même moitié de la portée.

Le fait de placer les notes dans des voix auxquelles sont attachés orientation de hampe et décalage, comme `\voiceOne` ou autre, peut permettre d'éviter ces avertissements.

Les notes se trouvant dans des voix au numéro plus élévé – `\voiceThree` ou `\voiceFour` – sont automatiquement décalées pour éviter que les empilements se chevauchent. Ceci aura pour résultat de visuellement décaler les notes affublées de hampe sans toutefois bouger les rondes, hormis dans le cas d'un réel chevauchement ou lorsque ces voix se croisent (`\voiceThree` au dessus de`\voiceOne`).

Voir aussi

Manuel d'initiation : Section "Instanciation explicite des voix" dans *Manuel d'initiation*, Section "Exemple concret" dans *Manuel d'initiation*.

Manuel de notation : Section "Polyphonie sur une portée" dans *Manuel de notation*, Section "Résolution des collisions" dans *Manuel de notation*.

2 Mise à jour avec `convert-ly`

Au fur et à mesure des améliorations de LilyPond, la syntaxe ou façon de libeller les fonctions et commandes peut être amenée à évoluer. Ceci peut avoir pour effet de générer des erreurs ou avertissements intempestifs, voire une sortie erronée, lorsque des fichiers créés avec une version antérieure sont traités par une version plus récente du progamme.

L'utilitaire `convert-ly` permet alors de mettre ces fichiers en conformité au fur et à mesure que de nouvelles versions de LilyPond sont disponibles.

2.1 LilyPond est une langue vivante

La syntaxe de LilyPond change de temps en temps, que ce soit pour rendre les fichiers plus faciles à lire et à écrire, ou pour intégrer de nouvelles fonctionnalités.

En voici un exemple flagrant :

Tous les noms des propriétés de `\paper` et `\layout` sont libellés sous la forme `premier-deuxième-troisième`. Nous avons constaté, une fois la version 2.11.60 mise à disposition, que la propriété `printallheaders` ne respectait pas cette convention. Aurions-nous dû la laisser telle que, au risque de dérouter les nouveaux utilisateurs par cette exception au formatage, ou bien la modifier – ce qui allait obliger ceux qui l'avaient déjà utilisée à se mettre en chasse ?

Pour ce cas d'espèce, nous avons décidé de changer le nom de cette propriété en `print-all-headers` et de permettre à ceux qui avaient utilisé l'ancienne syntaxe de modifier automatiquement leurs fichiers à l'aide de notre utilitaire `convert-ly`.

Malheureusement, `convert-ly` ne peut pas réaliser toutes les modifications. Par exemple, dans les versions de LilyPond antérieures à la 2.4.2, les accents et les lettres non anglaises étaient entrées en utilisant LaTeX – par exemple, `No\"el`. À partir de la version 2.6, le caractère `ë` doit être entré directement dans le fichier LilyPond comme caractère UTF-8. `convert-ly` ne peut pas changer tous les caractères LaTeX en caractères UTF-8 ; vous devez mettre à jour vos vieux fichiers LilyPond manuellement.

Les règles de conversion de `convert-ly` reposent sur la recherche et le remplacement de motifs textuels plutôt que sur les capacités intellectuelles de LilyPond, en conséquence de quoi :

- La fiabilité de la conversion dépend de la qualité même de chaque jeu de règles ainsi que sur la complexité des modifications respectives à apporter. Certaines conversions peuvent donc requérir une intervention manuelle ; la version de « départ » devrait toujours rester disponible pour comparaison.
- Seules des conversions à une syntaxe plus récente sont possibles ; aucune règle ne permet de revenir en arrière. La copie de travail d'un fichier LilyPond ne devrait donc être mise à jour que lorsque la version sur laquelle il repose n'est plus maintenue. Des système de gestion de version tels que Git permettent de se tenir à jour sur plusieurs versions.
- LilyPond, ainsi que Scheme, gèrent plutôt bien l'emplacement ou l'absence d'espaces ; les règles utilisées par `convert-ly` tendent cependant à effectuer certains postulats en matière de style. Suivre le style adopté dans les différent manuels est un gage de mise à jour sans problème si l'on considère que ces manuels sont eux-même mis à jour avec `convert-ly`.

2.2 Exécution de `convert-ly`

La commande `convert-ly` utilise les mentions de `\version` – que vous n'avez sûrement pas oublié de porter dans vos fichiers – pour déterminer le numéro de l'ancienne version. Mettre à jour votre fichier ne vous demande que de lancer

```
convert-ly -e monfichier.ly
```

dans le dossier où il se trouve. monfichier.ly sera mis à jour, avec un nouveau buméro en argument à \version, et vous aurez une copie de l'original : monfichier.ly~.

> **Note :** convert-ly effectuera les conversions jusqu'aux modifications de syntaxe les plus récentes qu'il contient. C'est la raison pour laquelle le numéro de \version modifié est la plupart du temps inférieur au propre numéro de version de convert-ly.

Vous pouvez convertir tous les fichiers d'un dossier en lançant

```
convert-ly -e *.ly
```

Les utilisatuers de Gnu/Linux ou de MacOS X peuvent lancer cette commande dans un terminal. Les utilisateurs de MacOS X disposent d'une entrée spécifique dans le menu : Compile > Update syntax.

Un utilisateur de Windows lancera la commande

```
convert-ly.py -e *.ly
```

dans l'interpréteur de commandes, qui se trouve normalement sous Démarrer > Accessoires > Interpréteur de commandes ou, pour la version 8, en faisant une recherche sur « interpréteur de commande ».

La conversion d'un jeu de fichiers répartis dans différents sous-répertoires s'obtient en lançant

```
find . -name '*.ly' -exec convert-ly -e '{}' \;
```

Ceci aura pour effet de rechercher et convertir tous les fichiers sources dans le répertoire en cours et dans tous ses sous-répertoires. les fichiers convertis se trouveront à leur emplacement d'orignie, tout comme les fichiers originels après renommage. Cette commande, bien qu'effective uniquement dans un terminal, devrait etre fonctnionnelle aussi pour les utilisateurs de MacOS X.

Les utilisateurs de windows utiliseront l'instruction

```
forfiles /s /M *.ly /c "cmd /c convert-ly.py -e @fichier"
```

Par ailleurs, il est possible de spécifier de manière explicite le chemin d'accès au dossier comportant des sous-répertoires où se trouvent les fichiers sources, à l'aide de l'option /p :

```
forfiles /s /p C:\Documents\MesPartitions /M *.ly /c "cmd /c convert-ly.py -e @fichier"
```

Dans le cas où ce chemin d'accès comporte des espaces, l'intégralité de ce chemin devra être borné par des guillemets informatiques :

```
forfiles /s /p "C:\Documents\Mes Partitions" /M *.ly /c "cmd /c convert-ly.py -e @fichi
```

2.3 Options en ligne de commande pour convert-ly

L'utilitaire convert-ly se lance de la manière suivante :

```
convert-ly [option]... fichier...
```

Vous pouvez utiliser les options :

-d, --diff-version-update
: actualise la valeur de \version, uniquement si le fichier a été effectivement modifié. Un numéro de version instable sera « arrondi » au niveau de la version stable suivante à moins que celui-ci ne soit supérieur à la version cible. En l'absence de cette option, ou bien si une conversion quelle qu'elle soit a modifié le fichier, la mention de version est porté à la valeur de la règle appliquée la plus récente.

`-e, --edit`
: pour éditer directement le fichier d'origine. Le fichier originel est renommé en que `monfichier.ly~`. Ce fichier de sauvegarde, selon le système d'exploitation, peut être « caché ».

  Vous pouvez aussi affecter un autre nom au fichier mis à jour et conserver votre fichier original en l'état :

  ```
  convert-ly monfichier.ly > monnouveaufichier.ly
  ```

  et, pour les utilisateurs de windows :

  ```
  convert-ly.py monfichier.ly > monnouveaufichier.ly
  ```

`-b, --backup-numbered`
: combine à l'option '`-e`', pour numéroter les sauvegardes de telle sorte qu'aucune version antérieure ne soit écrasée. Les fichiers de sauvegarde, selon le système d'exploitation, peuvent être « cachés ».

`-f, --from=`*`from-patchlevel`*
: pour définir le numéro de version à partir duquel vous voulez effectuer les conversions. Lorsque cette option n'est pas activée, `convert-ly` tentera de le déterminer sur la foi de la mention de `\version` contenue dans le fichier. Cette option s'utilise sous la forme : `--from=2.10.25`

`-h, --help`
: visualiser l'aide et quitter.

`-l` *`loglevel`*`, --loglevel=`*`loglevel`*
: pour régler le degré de verbosité à *loglevel*. Les différentes valeurs sont `NONE`, `ERROR`, `WARNING`, `PROGRESS` (par défaut) et `DEBUG`.

`-n, --no-version`
: Normalement, `convert-ly` ajoutera une indication de `\version` à votre fichier s'il n'en comporte pas. Cette option permet de passer outre.

`-s, --show-rules`
: pour afficher les conversions applicables.

`-t, --to=`*`to-patchlevel`*
: pour n'appliquer les conversions que jusqu'à une version déterminée. Il s'agit par défaut de la dernière version disponible. Le niveau demandé doit être supérieur à la version de départ.

  ```
  convert-ly --to=2.14.1 monfichier.ly
  ```

Lorsqu'il s'agit de fragments inclus dans un fichier texinfo, il vous faudra lancer

```
convert-ly --from=... --to=... --no-version *.itely
```

Lorsque vous désirez savoir quels changements de syntaxe sont intervenus entre deux versions de LilyPond, lancez

```
convert-ly --from=ancienne --to=récente -s
```

2.4 Problèmes d'exécution de `convert-ly`

Sous Windows, lorsque le nom du fichier original ou le chemin qui y mène comporte des espaces, l'interpréteur de commande requiert qu'il soit entouré de triples guillemets comme ci-dessous :

```
convert-ly """D:/Mes Partitions/Ode.ly""" > "D:/Mes Partitions/nouveau Ode.ly"
```

Lorsque la commande `convert-ly -e *.ly` échoue parce que son expansion dépasse la taille maximale d'une ligne, vous pouvez lancer `convert-ly` dans une boucle. L'exemple suivant permet, sous Unix, de convertir tous les fichiers `.ly` d'un même répertoire :

```
for f in *.ly; do convert-ly -e $f; done;
```

Avec l'interpréteur de commandes de Windows, la syntaxe consacrée est :

```
for %x in (*.ly) do convert-ly -e """%x"""
```

Toutes les évolutions du langage ne sont pas forcément prises en charge. `convert-ly` ne tolère qu'une seule option de sortie à la fois. La mise à jour automatique du code Scheme inclus dans les fichiers LilyPond est plus qu'hasardeuse ; attendez-vous à devoir mettre les mains dans le cambouis.

2.5 Conversions manuelles

En théorie, un programme tel que `convert-ly` devrait pouvoir traiter n'importe quel changement de syntaxe. En effet, si un programme informatique sait interpréter aussi bien une version que l'autre, un autre programme informatique doit alors être capable de traduire un fichier donné[1].

Le projet LilyPond ne dispose cependant que de ressources limitées : les conversions ne sont pas toutes automatisées. Voici une liste de problèmes clairement identifiés :

```
1.6->2.0:
 Doesn't always convert figured bass correctly, specifically things like {<
>}.  Mats' comment on working around this:
   To be able to run convert-ly
   on it, I first replaced all occurrences of '{<' to some dummy like '{#'
   and similarly I replaced '>}' with '&}'.  After the conversion, I could
   then change back from '{ #' to '{ <' and from '& }' to '> }'.
 Doesn't convert all text markup correctly.  In the old markup syntax,
 it was possible to group a number of markup commands together within
parentheses, e.g.
   -#'((bold italic) "string")
   This will incorrectly be converted into
   -\markup{{\bold italic} "string"}
   instead of the correct
   -\markup{\bold \italic "string"}
2.0->2.2:
 Doesn't handle \partcombine
 Doesn't do \addlyrics => \lyricsto, this breaks some scores with multiple
stanzas.
2.0->2.4:
 \magnify isn't changed to \fontsize.
    - \magnify #m => \fontsize #f, where f = 6ln(m)/ln(2)
 remove-tag isn't changed.
    - \applyMusic #(remove-tag '. . .) => \keepWithTag #'. . .
 first-page-number isn't changed.
    - first-page-number no => print-first-page-number = ##f
 Line breaks in header strings aren't converted.
    - \\\\  as line break in \header strings => \markup \center-align <
      "First Line" "Second Line" >
 Crescendo and decrescendo terminators aren't converted.
    - \rced => \!
    - \rc => \!
2.2->2.4:
```

[1] Ceci est réalisable tant que le fichier LilyPond ne contient pas de Scheme. Dès lors qu'un fichier contient du Scheme, des bribes de langage évolué se retrouvent danas le fichier LilyPond, ce qui conduit immanquablement au « problème de l'arrêt » bien connu en informatique.

```
 \turnOff (used in \set Staff.VoltaBracket = \turnOff) is not properly
converted.
2.4.2->2.5.9
 \markup{ \center-align <{ ... }> } should be converted to:
 \markup{ \center-align {\line { ... }} }
 but now, \line is missing.
2.4->2.6
 Special LaTeX characters such as $~$ in text are not converted to UTF8.
2.8
 \score{} must now begin with a music expression.  Anything else
 (particularly \header{}) must come after the music.
```

3 Association musique-texte avec lilypond-book

Vous pouvez inclure des partitions dans un document tout comme vous feriez pour n'importe quelle image. Ces images sont générées séparément – que ce soit sous forme de description PostScript ou au format PNG – puis incluses dans votre document LaTeX ou HTML.

lilypond-book permet d'automatiser ces opérations : le programme extrait de votre document les fragments de musique, les traite grâce à lilypond puis en restitue la partition dans votre document. Largeur de ligne et taille de la fonte sont adaptées pour correspondre à la mise en forme de votre document.

lilypond-book est un script indépendant de lilypond et se lance en ligne de commande – pour plus de précisions, consultez Section 1.2 [Utilisation en ligne de commande], page 1. Si vous utilisez MacOS ou Windows et rencontrez quelque difficulté avec lilypond-book, référez-vous à Section “MacOS X” dans *Informations générales* ou Section “Windows” dans *Informations générales*.

lilypond-book s'applique aux documents LaTeX, HTML, Texinfo et DocBook.

3.1 Exemple de document musicologique

Un certain nombre d'ouvrages peuvent être illustrés par des extraits musicaux, qu'il s'agisse d'un traité de musicologie, d'un carnet de chant ou d'un manuel à l'exemple de celui que vous consultez actuellement. Cet agencement peut se faire « à la main » par importation d'un graphique PostScript dans le traitement de texte. Les développeurs de LilyPond ont cependant créé un outil permettant d'automatiser ces opérations pour ce qui concerne les documents HTML, LaTeX, Texinfo et DocBook.

Un script – lilypond-book – se charge d'extraire les fragments de musique, puis de les mettre en forme avant de renvoyer la « partition » correspondante. Voici un court exemple utilisable avec LaTeX. Dans la mesure où il est suffisamment parlant, nous nous abstiendrons de le commenter.

Fichier d'entrée

```
\documentclass[a4paper]{article}

\begin{document}

Un document destiné à être traité par \verb+lilypond-book+ peut tout à
fait mélanger de la musique et du texte.
Par exemple,

\begin{lilypond}
\relative {
  c'2 e2 \tuplet 3/2 { f8 a b } a2 e4
}
\end{lilypond}

Les options sont indiquées entre crochets.

\begin{lilypond}[fragment,quote,staffsize=26,verbatim]
  c'4 f16
\end{lilypond}

Des extraits plus conséquents peuvent faire l'objet d'un fichier
```

```
indépendant, alors inclus avec \verb+\lilypondfile+.

\lilypondfile[quote,noindent]{screech-and-boink.ly}

(Si besoin, remplacez @file{screech-and-boink.ly} par
n'importe quel fichier @file{.ly} qui se trouve dans
le même répertoire que le présent fichier.)

\end{document}
```

Traitement

Enregistrez ces lignes dans un fichier nommé `lilybook.lytex` puis, dans un terminal, lancez

```
lilypond-book --output=out --pdf lilybook.lytex
lilypond-book (GNU LilyPond) 2.19.31
Lecture de lilybook.lytex...
...nous vous épargnons le verbiage de la console...
Compilation de lilybook.tex...
cd out
pdflatex lilybook
...nous vous épargnons le verbiage de la console...
xpdf lilybook
(remplacez xpdf par votre lecteur de PDF habituel)
```

Le traitement par `lilypond-book` puis `latex` va générer un certain nombre de fichiers temporaires susceptibles d'encombrer inutilement votre répertoire de travail, aussi nous vous recommandons d'utiliser l'option `--output=`*`répertoire`* afin que les fichiers créés soient isolés dans le sous-répertoire `répertoire`.

Pour terminer, voici le résultat de cet exemple pour LaTeX.[1]

[1] Ce manuel étant réalisé avec Texinfo, il se peut que la mise en forme diverge quelque peu.

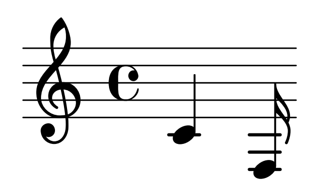
Résultat

Un document destiné à être traité par `lilypond-book` peut tout à fait mélanger de la musique et du texte. Par exemple,

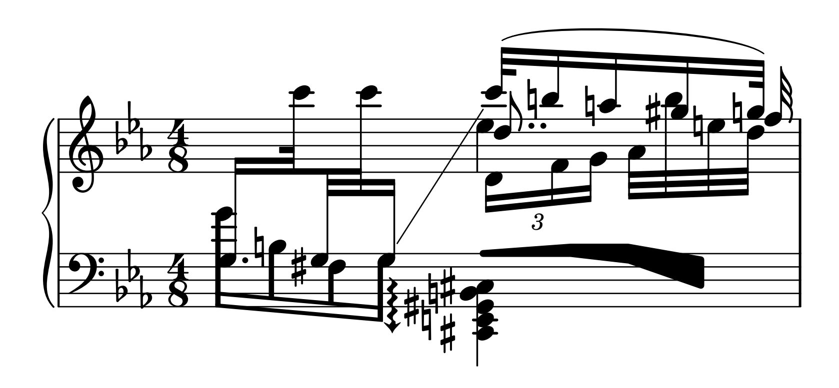


Les options sont indiquées entre crochets.

```
c'4 f16
```
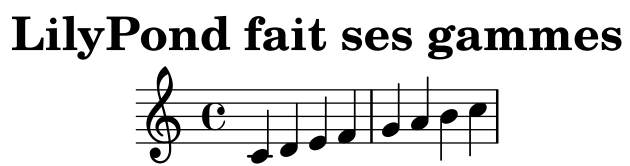


Des extraits plus conséquents peuvent faire l'objet d'un fichier indépendant, alors inclus avec `\lilypondfile`.



Lorsque vous désirez y inclure un `tagline`, personnalisé ou non, l'intégralité de l'extrait devra apparaître dans une construction de type `\book { }`.

```
\book{
  \header{
    title = "LilyPond fait ses gammes"
  }

  \relative {
    c' d e f g a b c
  }
}
```



Music engraving by LilyPond 2.19.31—www.lilypond.org

3.2 Association musique-texte

Nous allons nous intéresser, dans les lignes qui suivent, à la manière d'intégrer LilyPond selon différents types de format.

3.2.1 LaTeX

LaTeX peut être considéré comme le standard de publication dans le domaine des sciences exactes. Il repose sur le moteur typographique TeX, le *nec plus ultra* en la matière.

Consultez *The Not So Short Introduction to LaTeX* en français (`http://www.ctan.org/tex-archive/info/lshort/french/`) pour un aperçu des possibilités de LaTeX.

Afin d'insérer de la musique dans vos fichiers LaTeX, `lilypond-book` dispose des environnements et commandes suivantes :

- la commande `\lilypond{...}` qui permet de directement saisir du code LilyPond simple ;
- l'environnement `\begin{lilypond}...\end{lilypond}` qui permet de saisir directement du code LilyPond plus élaboré ;
- la commande `\lilypondfile{...}` qui permet d'insérer un fichier LilyPond ;
- la commande `\musicxmlfile{...}` qui permet d'insérer un fichier MusicXML qui sera alors traité par `musicxml2ly` puis `lilypond`.

Il suffit, pour inclure de la musique, d'utiliser l'une des instructions suivantes :

```
\begin{lilypond}[liste,des,options]
  VOTRE CODE LILYPOND
\end{lilypond}

\lilypond[liste,des,options]{ VOTRE CODE LILYPOND }

\lilypondfile[liste,des,options]{fichier}

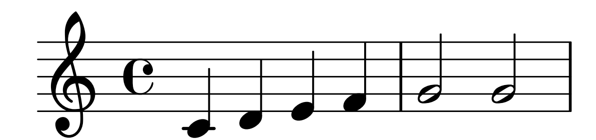
\musicxmlfile[liste,des,options]{fichier}
```

Par ailleurs, la commande `\lilypondversion` vous permet d'afficher le numéro de version de LilyPond. Lancer `lilypond-book` produira un fichier qui sera ensuite traité par LaTeX.

Voici quelques exemples. L'environnement `lilypond`

```
\begin{lilypond}[quote,fragment,staffsize=26]
  c' d' e' f' g'2 g'2
\end{lilypond}
```
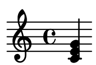
produit



La version abrégée

```
\lilypond[quote,fragment,staffsize=11]{<c' e' g'>}
```

produit



Dans l'état actuel des choses, il n'est pas possible d'inclure des accolades – `{` ou `}` – dans un `\lilypond{}` ; cette commande n'est donc pertinente que lorsque conjuguée à l'option `fragment`.

La longueur par défaut des portées sera ajustée en fonction des commandes contenues dans le préambule du document – ce qui précède la ligne `\begin{document}`. La commande `lilypond-book` les transmet à LaTeX afin de connaître la largeur du texte, et par voie de conséquence déterminer la longueur des portées. Notez bien que cet algorithme heuristique n'est pas infaillible ; vous devrez alors recourir à l'option `line-width`.

Dès lors qu'elles auront été définies dans votre document, les macros suivantes seront appelées avant chaque extrait musical :

- `\preLilyPondExample` avant la musique,
- `\postLilyPondExample` après la musique,
- `\betweenLilyPondSystem[1]` entre les systèmes, si tant est que `lilypond-book` a découpé la partition en plusieurs fichiers PostScript. Elle requiert un paramètre et reçoit le nombre de fichiers inclus dans l'extrait. Par défaut, elle insère simplement un `\linebreak`.

Morceaux choisis

Lorsque, pour les besoins de la démonstration, certains éléments musicaux tels que des liaisons – de phrasé ou de prolongation – continuent après le fragment qui vous intéresse, il suffit d'insérer un saut de ligne et de limiter le nombre de systèmes à inclure.

En ce qui concerne LaTeX, vous devrez définir `\betweenLilyPondSystem` de telle sorte que l'inclusion cesse dès que le nombre de systèmes requis est atteint. Dans la mesure où `\betweenLilyPondSystem` n'est appelé qu'**après** le premier système, inclure un seul système est un jeu d'enfant :

```
\def\betweenLilyPondSystem#1{\endinput}

\begin{lilypond}[fragment]
  c'1\( e'( c'~ \break c' d) e f\)
\end{lilypond}
```

Pour un plus grand nombre de systèmes, il suffit d'insérer un test conditionnel TeX avant le `\endinput`. À partir de l'exemple qui suit, remplacez le « 2 » par le nombre de systèmes dont vous aurez besoin :

```
\def\betweenLilyPondSystem#1{
    \ifnum##1<2\else\expandafter\endinput\fi
}
```

Étant donné que `\endinput` arrête immédiatement le traitement du fichier source en cours, l'insertion du `\expandafter` permet de repousser ce `\endinput` après le `\fi` ; la clause `\if-\fi` sera alors respectée.

Gardez à l'esprit que `\betweenLilyPondSystem` est effectif tant que TeX n'est pas sorti du groupe en cours – tel que l'environnement LaTeX – ou écrasé par une nouvelle définition pour la suite du document la plupart du temps. Pour réinitialiser cette définition, insérez

```
\let\betweenLilyPondSystem\undefined
```

dans votre document LaTeX.

La création d'une macro TeX permet de se simplifier la vie :

```
\def\onlyFirstNSystems#1{
    \def\betweenLilyPondSystem##1{%
      \ifnum##1<#1\else\expandafter\endinput\fi}
}
```

Il suffit alors, avant chacun des fragments à inclure, de spécifier le nombre de systèmes requis :

```
\onlyFirstNSystems{3}
\begin{lilypond}...\end{lilypond}
```

```
\onlyFirstNSystems{1}
\begin{lilypond}...\end{lilypond}
```

Voir aussi

lilypond-book dispose d'options en ligne de commande particulières. Elles sont consultables, ainsi que d'autres détails spécifiques au traitement de documents LaTeX, au chapitre Section 3.4 [Utilisation de lilypond-book], page 33.

3.2.2 Texinfo

Texinfo est le format standard pour toute la documentation du projet GNU. À titre d'exemple, toute la documentation de LilyPond – qu'il s'agisse des versions HTML, PDF ou info – est générée à partir de documents Texinfo.

Afin d'insérer de la musique dans vos fichiers Texinfo, lilypond-book dispose des environnements et commandes suivantes :

- la commande `@lilypond{...}` qui permet de directement saisir du code LilyPond simple ;
- l'environnement `@lilypond...@end lilypond` qui permet de saisir directement du code LilyPond plus élaboré ;
- la commande `@lilypondfile{...}` qui permet d'insérer un fichier LilyPond ;
- la commande `@musicxmlfile{...}` qui permet d'insérer un fichier MusicXML qui sera alors traité par `musicxml2ly` puis `lilypond`.

Il suffit, pour inclure de la musique, d'utiliser l'une des instructions suivantes dans votre fichier source :

```
@lilypond[liste,des,options]
  VOTRE CODE LILYPOND
@end lilypond

@lilypond[liste,des,options]{ VOTRE CODE LILYPOND }

@lilypondfile[liste,des,options]{fichier}

@musicxmlfile[liste,des,options]{fichier}
```

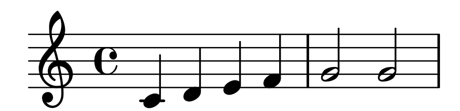
Par ailleurs, l'utilisation d'un `@lilypondversion` permet d'afficher la version de LilyPond utilisée.

Le traitement du fichier source par lilypond-book génère un fichier Texinfo (extension `.itexi`) qui contiendra les balises `@image` pour les formats HTML, Info ou imprimable. Les images générées par lilypond-book sont au format EPS et PDF en vue d'une impression, et au format PNG pour leur utilisation en HTML ou Info.

En voici deux exemples. Un environnement `lilypond`

```
@lilypond[fragment]
c' d' e' f' g'2 g'
@end lilypond
```

produit



La version abrégée

```
@lilypond[fragment,staffsize=11]{<c' e' g'>}
```

produit



Contrairement à LaTeX, @lilypond{...} ne va pas intégrer d'image dans le document, mais générer un paragraphe.

3.2.3 HTML

Afin d'insérer de la musique dans vos fichiers HTML, lilypond-book dispose des environnements et commandes suivantes :

- la commande <lilypond ... /> qui permet de directement saisir du code LilyPond simple ;
- l'environnement <lilyond>...</lilypond> qui permet de saisir directement du code LilyPond plus élaboré ;
- la commande <lilypondfile>...</lilypondfile> qui permet d'insérer un fichier LilyPond ;
- la commande <musicxmlfile>...</musicxmlfile> qui permet d'insérer un fichier MusicXML qui sera alors traité par musicxml2ly puis lilypond.

Il suffit, pour inclure de la musique, d'utiliser l'une des instructions suivantes dans votre fichier source :

```
<lilypond liste des options>
  VOTRE CODE LILYPOND
</lilypond>
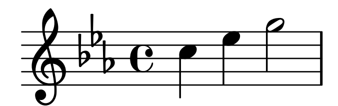
<lilypond liste des options: VOTRE CODE LILYPOND />

<lilypondfile liste des options>fichier</lilypondfile>

<musicxmlfile liste des options>fichier</musicxmlfile>
```

Il suffit, pour inclure de la musique, d'utiliser

```
<lilypond fragment relative=2>
\key c \minor c4 es g2
</lilypond>
```

lilypond-book produira alors un fichier HTML contenant les balises d'image pour les fragments de musique :



Pour insérer l'image au fil du texte, il suffit d'utiliser <lilypond ... />, tout en séparant options et musique par un caractère deux points, comme ici :

```
De la musique <lilypond relative=2: a b c/> au milieu d'une ligne de texte.
```

Lorsque l'inclusion concerne des fichiers indépendants, utilisez

```
<lilypondfile option1 option2...>fichier</lilypondfile>
```

La syntaxe pour <musicXmlfile> est identique à celle de <lilypondfile>, à ceci près qu'elle fait référence à un fichier MusicXML plutôt qu'à un fichier LilyPond.

Une liste des différentes options utilisables avec les balises `lilypond` et `lilypondfile` est disponible, à la rubrique Section 3.3 [Options applicables aux fragments de musique], page 31.

Par ailleurs, la commande `<lilypondversion/>` vous permet d'afficher le numéro de version de LilyPond.

3.2.4 DocBook

L'inclusion de documents LilyPond ne doit nuire en rien à la conformité du document DocBooK ; l'utilisation d'éditeurs spécialisés ainsi que d'outils de validation en sera ainsi préservée. C'est la raison pour laquelle nous ne définirons pas de balise spécifique ; nous respecterons plutôt les conventions des éléments standard de DocBook.

Conventions communes

Quel que soit le type d'extrait à inclure, nous utiliserons les éléments `mediaobject` et `inlinemediaobject`, de telle sorte que ces inclusions soient incorporées directement ou non dans le document final. Les options de formatage des extraits en question sont fournies par la propriété `role` de l'élément central – voir les paragraphes suivants. Les balises sont déterminées de manière à ce que les éditeurs DocBook prennent en charge du mieux possible leur contenu. Les fichiers DocBook destinés à un traitement par `lilypond-book` doivent avoir une extension `.lyxml`.

Inclusion d'un fichier LilyPond

Il s'agit en fait du cas le plus simple. Le fichier à inclure doit avoir une extension `.ly` et sera inséré comme n'importe quel `imageobject`, en respectant la structure suivante :

```
<mediaobject>
  <imageobject>
    <imagedata fileref="music1.ly" role="printfilename" />
  </imageobject>
</mediaobject>
```

Vous pouvez utiliser, en tant que balise externe, aussi bien `mediaobject` que `inlinemediaobject`.

Inclusion de code LilyPond

L'inclusion de code LilyPond se réalise à l'aide d'un `programlisting` auquel on associe le langage `lilypond`. En voici la syntaxe :

```
<inlinemediaobject>
  <textobject>
    <programlisting language="lilypond" role="fragment verbatim staffsize=16 ragged-right re
\context Staff \with {
  \remove "Time_signature_engraver"
  \remove "Clef_engraver"}
  { c4( fis) }
    </programlisting>
  </textobject>
</inlinemediaobject>
```

Comme vous le remarquez, la balise externe – qu'il s'agisse d'un `mediaobject` ou d'un `inlinemediaobject` – comporte un bloc `textobject` qui contiendra le `programlisting`.

Génération du document DocBook

`lilypond-book` génère, à partir d'un fichier `.lyxml`, un document DocBook tout à fait valide – extension `.xml` – que vous pourrez ensuite traiter avec votre application habituelle. Dans le

cas de dblatex (`http://dblatex.sourceforge.net`), vous obtiendrez alors automatiquement un fichier PDF. Les feuilles de style XSL DocBook officielles permettent de générer du HTML (HTML Help, JavaHelp, etc.) ; vous pourriez néanmoins devoir y apporter quelques adaptations.

3.3 Options applicables aux fragments de musique

Dans les lignes qui suivent, l'appellation « commande LilyPond » fait référence à toutes celles vues plus haut et qui font appel à `lilypond-book` pour produire un extrait musical. Pour plus de simplicité, nous ne parlerons que de la syntaxe applicable à LaTeX.

Nous attirons votre attention sur le fait que les différentes options sont lues de la gauche vers la droite. Si une option est transmise plusieurs fois, seule la dernière sera prise en compte.

Les commandes LilyPond acceptent les options suivantes :

`staffsize=`*`hauteur`*
: Définit la taille de portée à *hauteur* exprimée en points.

`ragged-right`
: Produit des lignes en pleine largeur avec un espacement naturel. En d'autres termes, sera ajoutée la commande de mise en forme `ragged-right = ##t`. Il s'agit de l'option par défaut de la commande `\lilypond{}` en l'absence d'option `line-width`. C'est aussi l'option par défaut pour l'environnement `lilypond` lorsque l'option `fragment` est activée sans avoir défini explicitement de longueur de ligne.

`noragged-right`
: Dans le cas où l'extrait tient sur une seule ligne, la portée sera étirée pour correspondre à la longueur de ligne du texte. Autrement dit, la commande de mise en forme `ragged-right = ##f` s'ajoute à l'extrait LilyPond.

`line-width`
`line-width=`*`taille`*`\`*`unité`*
: Détermine la longueur de ligne à *taille*, exprimée en *unité*. *unité* peut prendre les valeurs `cm`, `mm`, `in` ou `pt`. Cette option n'affectera que le résultat de LilyPond – la longueur de la portée – et en aucun cas la mise en forme du texte.

: En l'absence d'argument, la longueur de ligne sera définie à une valeur par défaut telle que calculée par un algorithme heuristique.

: Lorsque l'option `line-width` n'est pas utilisée, `lilypond-book` tentera de déterminer des valeurs par défaut pour les cas où les environnements `lilypond` ne font pas appel à `ragged-right`.

`papersize=`*`chaîne`*
: Détermine le format du papier à *chaîne* tel que défini dans le fichier `scm/paper.scm` – `a5`, `quarto`, `11x17`, etc.

: Toute valeur non mentionnée dans `scm/paper.scm` sera rejetée. `lilypond-book` émettra un message d'avertissement et l'extrait utilisera le format par défaut, à savoir `a4`.

`notime`
: Désactive l'impression des métriques et barres de mesure pour l'intégralité de la partition.

`fragment`
: Laisse à `lilypond-book` le soin d'ajouter ce qui est indispensable, de telle sorte que vous pouvez vous contenter d'un

```
c'4
```

: sans `\layout`, `\score`, etc.

`nofragment`
: N'ajoute rien à ce qui se trouve dans l'environnement LilyPond. À noter qu'il s'agit de l'option par défaut.

`indent=`*`taille`*`\`*`unité`*
: Définit l'indentation du premier système à *taille*, exprimée en *unité* – `cm`, `mm`, `in` ou `pt`. Cette option n'affecte que LilyPond, et en aucun cas la mise en forme du texte.

`noindent`
: Ramène l'indentation du premier système à zéro. Cette option n'affecte que LilyPond, et en aucun cas la mise en forme du texte. Dans la mesure où il s'agit du comportement par défaut, point n'est besoin de spécifier `noindent`.

`quote`
: Réduit la longueur des lignes musicales de $2 * 0.4\,\text{in}$ (soit $2 \ * \ 10,16 \ \text{mm}$) pour renvoyer l'extrait dans un bloc de citation. La valeur « 0,4 pouce » est contrôlée par l'option `exampleindent`.

`exampleindent`
: Détermine la valeur de l'indentation qui sera utilisée par l'option `quote`.

`relative`
`relative=`*`n`*
: Utilise le mode d'octave relative. Les notes sont donc par défaut positionnées relativement au do central. L'argument – un nombre entier – fourni à l'option `relative` spécifie l'octave de départ de l'extrait ; `1` correspond au do central. Cette option `relative` n'a d'effet que si elle est utilisée en combinaison avec l'option `fragment` ; autrement dit, l'option `fragment` est implicite dès lors que `relative` est explicité.

La documentation de LilyPond, comme nous l'avons déjà vu, use abondamment de `lilypond-book`. Elle utilise à cet effet quelques options particulières.

`verbatim`
: L'argument de la commande LilyPond est recopié textuellement dans le fichier généré, avant l'image de la partition. Cependant, cette option n'est pas pleinement opérationnelle lorsqu'un `\lilypond{}` se trouve au milieu d'un paragraphe.

L'utilisation conjointe d'un `verbatim` et de la commande `lilypondfile` permet de n'inclure textuellement qu'une seule partie du fichier source. `lilypond-book` reproduira alors textuellement la partie du fichier source comprise entre les commentaires `begin verbatim` et éventuellement `end verbatim`. Si l'on considère le fichier source suivant, la musique sera interprétée en mode relatif, mais la recopie du code ne comportera pas l'assertion du bloc `relative` :

```
\relative { % begin verbatim
  c'4 e2 g4
  f2 e % end verbatim
}
```

donnera dans un bloc *verbatim* précédant la partition :

```
  c4 e2 g4
  f2 e
```

Si d'aventure vous désirez traduire les commentaires et noms de variable dans le rendu textuel plutôt que dans le fichier source, vous devrez définir la variable d'environnement `LYDOC_LOCALEDIR` qui pointera vers un répertoire contenant l'arborescence des catalogues de messages – fichiers d'extension `.mo` – du domaine `lilypond-doc`.

`addversion`
: Cette option, effective uniquement avec Texinfo, permet d'ajouter une ligne `\version @w{"@version{}"}` au `verbatim`.

`texidoc`
Option disponible uniquement avec Texinfo.
Dès lors qu'un fichier `toto.ly` contient dans sa section `\header` un champ `texidoc`, l'appel de `lilypond` avec l'option `--header=texidoc` créera le fichier `toto.texidoc`. Par ailleurs, c'est le contenu de ce `toto.texidoc` qui sera ensuite recopié par `lilypond-book` en préambule de l'extrait de partition – soit avant l'environnement `example` créé par un `quote`.

Prenons par exemple le fichier `toto.ly` dont le contenu est

```
\header {
  texidoc = "This file demonstrates a single note."
}
{ c'4 }
```

et quelque part dans notre document Texinfo `test.texinfo`

```
@lilypondfile[texidoc]{toto.ly}
```

La ligne de commande suivante produira le résultat escompté.

```
lilypond-book --pdf --process="lilypond \
  -dbackend=eps --header=texidoc" test.texinfo
```

La plupart des fichiers de test contenus dans le répertoire `input` de la distribution est constituée de la sorte.

Cette option est fort utile dans le cadre de l'adaptation en langue étrangère. En effet, s'il est spécifié dans le document Texinfo une clause `@documentlanguage` *`LANGUE`*, la présence d'une variable `texidoc`*`LANGUE`* dans l'entête du fichier `toto.ly` entraînera la reproduction – par l'appel `lilypond --header=texidoc`*`LANGUE`* – du contenu de `toto.texidoc`*`LANGUE`* en lieu et place de celui de `toto.texidoc`.

`doctitle`
Option disponible uniquement avec Texinfo.
Cette option fonctionne selon le même principe que l'option `texidoc` : lorsqu'un fichier `toto.ly` contient dans son `\header` une variable `doctitle` et que `lilypond` est appelé avec l'option `doctitle`, le contenu de cette variable – une simple ligne de *texte* – sera recopié dans un fichier `toto.doctitle` puis inséré dans le document Texinfo sous la forme `@lydoctitle` *`texte`*. `@lydoctitle` doit faire l'objet d'une macro, définie dans le document Texinfo.

Il en va de l'option `doctitle` comme de l'option `texidoc` en matière d'adaptation en langue étrangère.

`nogettext`
Option disponible uniquement pour Texinfo.
Commentaires et noms de variable ne seront pas traduits dans la recopie textuelle du code.

`printfilename`
Lorsqu'un fichier source LilyPond est inclus à l'aide de `\lilypondfile`, le nom du fichier sera reproduit juste au dessus de l'extrait. Si le résultat est un fichier HTML, il s'agira alors d'un lien. Seul le nom du fichier est imprimé ; autrement dit, le chemin d'accès au fichier est tronqué.

3.4 Utilisation de `lilypond-book`

`lilypond-book` produit un fichier qui aura, selon le format de sortie spécifié, l'extension `.tex`, `.texi`, `.html` ou `.xml`. Les fichiers `.tex`, `.texi` et `.xml` nécessitent un traitement complémentaire.

Instructions spécifiques à certains formats

LaTeX

Un document LaTeX destiné à l'impression ou à la publication peut se traiter de deux manières différentes : générer directement un PDF à l'aide de PDFLaTeX, ou bien générer un fichier avec LaTeX qui sera ensuite passé à un traducteur DVI-PostScript comme `dvips`. La première façon est de loin la plus simple et c'est celle que nous vous recommandons[1] ; quelque soit votre préférence, sachez que vous pouvez aller du PostScript au PDF avec des outils tels que `ps2pdf` et `pdf2ps` – tous deux inclus dans la distribution de Ghostscript.

La production d'un PDF avec PDFLaTeX se fait en lançant les commandes

```
lilypond-book --pdf monfichier.lytex
pdflatex monfichier.tex
```

La séquence LaTeX/`dvips`/`ps2pdf` suivante permet de produire un PDF :

```
lilypond-book monfichier.lytex
latex monfichier.tex
dvips -Ppdf monfichier.dvi
ps2pdf monfichier.ps
```

Le fichier `.dvi` généré lors de ce traitement ne contient aucune tête de note, ce qui est tout à fait normal ; elles seront incluses lors de la génération du `.ps` puis dans le `.pdf`.

La commande `dvips` peut déclencher certains messages concernant des fontes, que vous pouvez ignorer sans scrupule.
Si vous utilisez `latex` en mode colonnage, n'oubliez pas d'ajouter `-t landscape` aux options de `dvips`.

Les environnements tels que

```
\begin{lilypond} ... \end{lilypond}
```

ne sont pas interprétés par LaTeX. En fait, `lilypond-book` extrait ces « environnements » dans des fichiers accessoires et les traite par LilyPond. Il récupère ensuite les graphiques résultants et crée un fichier `.tex` dans lequel les macros `\begin{lilypond}...\end{lilypond}` sont alors remplacées par des commandes « d'inclusion de graphique ». C'est seulement à ce moment là que LaTeX est lancé – bien que LaTeX aura préalablement tourné, cela aura été en fait sur un document « vide » et pour calculer certains éléments comme `\linewidth`.

Problèmes connus et avertissements

La commande `\pageBreak` est inopérante dans un environnement `\begin{lilypond}...\end{lilypond}`.

Il en va de même pour un certain nombre de variables appartenant au bloc `\paper`. Utilisez, entre autres, un `\newcommand` avec la macrocommande `\betweenLilyPondSystem` dans le préambule.

```
\newcommand{\betweenLilyPondSystem}[1]{\vspace{36mm}\linebreak}
```

Texinfo

La génération d'un document Texinfo – quel que soit le format final – s'obtient grâce aux commandes Texinfo habituelles, c'est à dire `texi2pdf`, `texi2dvi` ou `makeinfo` selon le résultat que vous désirez obtenir. Pour plus de détails, consultez la documentation de Texinfo.

Options en ligne de commande

`lilypond-book` accepte les options suivantes :

---

[1] Sachant que vous ne disposez pas forcément de PDFLaTeX et LaTeX pour compiler un document LaTeX, nous vous présentons les deux méthodes.

`-f` *format*
`--format=`*format*
: Spécifie le type de document à traiter : `html`, `latex`, `texi` (valeur par défaut) ou `docbook`. Lorsque cette option n'est pas mentionnée, `lilypond-book` tente de déterminer automatiquement le format – voir Section 3.5 [Extensions de nom de fichier], page 37. À l'heure actuelle, `texi` est équivalant à `texi-html`.

`-F` *filtre*
`--filter=`*filtre*
: Passe les extrait au travers de *filtre* avant de traiter le fichier. Cette option permet de, par exemple, appliquer les mises à jour de LilyPond aux extraits avant de traiter le fichier :

```
lilypond-book --filter='convert-ly --from=2.0.0 -' mon-book.tely
```

`-h`
`--help` Affiche un bref résumé des options.

`-I` *dir*
`--include=`*répertoire*
: Ajoute *répertoire* au chemin des inclusions. Si des extraits ont déjà été compilés dans l'un des répertoires inclus, `lilypond-book` ne les réécrira pas dans le répertoire de sortie ; il sera donc nécessaire, dans la suite du traitement par `makeinfo` ou `latex`, de penser à utiliser cette même option `-I` *répertoire*.

`-l` *loglevel*
`--loglevel=`*loglevel*
: Détermine le degré de verbosité à *loglevel*. Les différentes valeurs admises sont `NONE`, `ERROR`, `WARNING`, `PROGRESS` (par défaut) et `DEBUG`. Lorsque cette option n'est pas activée, c'est le niveau déterminé par la variable d'environnement `LILYPOND_BOOK_LOGLEVEL` qui sera utilisé.

`-o` *dir*
`--output=`*répertoire*
: Regroupe les fichiers générés dans *répertoire*. `lilypond-book` crée un certain nombre de fichiers à l'usage de LilyPond. Afin d'éviter de polluer votre répertoire source, nous vous conseillons d'utiliser l'option `--output`, puis de vous rendre dans ce répertoire pour y lancer les commandes `latex` ou `makeinfo`.

```
lilypond-book --output=out monfichier.lytex
cd out
...
```

`--skip-lily-check`
: Désactive la mise en échec en l'absence de sortie de lilypond.
Option utilisée pour la documentation au format Info sans images.

`--skip-png-check`
: Désactive la mise en échec en l'absence d'images PNG correspondant aux fichiers EPS.
Option utilisée pour la documentation au format Info sans images.

`--lily-output-dir=`*rép*
: Écrit les fichiers lily-XXX dans *rép* et crée un lien vers le répertoire spécifié par `--output`. Cette option permet d'économiser du temps lors de la génération de documents qui se trouvent dans différents répertoires et partagent un certain nombre d'extraits identiques.

`--lily-loglevel=`*loglevel*
: Détermine le degré de verbosité lors des appels à `lilypond`. Les valeurs autorisée de *loglevel* sont : `NONE`, `ERROR`, `WARNING`, `BASIC_PROGRESS`, `PROGRESS`, `INFO` (par défaut) et `DEBUG`. Lorsque cette option n'est pas activée, c'est le niveau déterminé par la variable d'environnement `LILYPOND_LOGLEVEL` qui sera utilisé.

`--info-images-dir=`*répertoire*
: Formate la sortie Texinfo de telle sorte que Info cherche les images de musique dans *répertoire*.

`--latex-program=`*programme*
: Utilise l'exécutable `programme` en lieu et place de `latex`. C'est l'option que vous utiliserez si vous préférez `xelatex` par exemple.

`--left-padding=`*distance*
: Décale les figures EPS de *distance* – exprimée en millimètres (3 par défaut). Cette option est utile lorsque les lignes de musique débordent sur la marge droite.

  Rappelez-vous que la largeur d'un système dépend des élément contenus dans sa marge gauche, tels que numéro de mesure et nom d'instrument. Cette option permet de « raccourcir » les lignes et de les décaler vers la droite, de la distance donnée en argument.

`-P` *commande*
`--process=`*commande*
: Traite les extraits LilyPond avec *commande*. Par défaut, il s'agit de `lilypond`. Rappelez-vous que `lilypond-book` ne peut en même temps traiter l'option `--filter` et l'option `--process`.

`--pdf`
: Crée des fichiers PDF pour les retraiter avec PDFLaTeX.

`--redirect-lilypond-output`
: Le résultat des commandes est habituellement affiché dans le terminal. Cette option permet de rediriger tout le verbiage dans un journal situé dans le même répertoire que le fichier source.

`--use-source-file-names`
: Cette option permet d'affecter aux fichiers correspondant aux extraits de musique le même nom que leur source. Elle n'est fonctionnelle que dans le cas où la partition est incluse à l'aide de `lilypondfile`, et que les répertoires mentionnés par les options `--output-dir` et `--lily-output-dir` diffèrent.

`-V`
`--verbose`
: `lilypond-book` sait être volubile ! Cette option est équivalente à `--loglevel=DEBUG`.

`-v`
`--version`
: Affiche le numéro de version.

Problèmes connus et avertissements

`lilypond-book` ne sait pas interpréter la commande Texinfo `@pagesize`. Dans le même ordre d'idée, des commandes LaTeX modifiant les marges et longueur de ligne mentionnées après le préambule seront ignorées.

Lorsqu'une section LilyPond contient plusieurs `\score`, seul le premier sera traité.

3.5 Extensions de nom de fichier

Vous pouvez affecter à votre fichier source n'importe quelle extension. Nous vous recommandons cependant un certain nombre d'extensions selon le format de sortie désiré – voir Section 3.4 [Utilisation de lilypond-book], page 33. Une extension hors du commun vous obligera à spécifier le format de sortie, alors que `lilpond-book` est en mesure de déterminer le format de sortie en fonction de l'extension du fichier source.

extension	**format résultant**
`.html`	HTML
`.htmly`	HTML
`.itely`	Texinfo
`.latex`	LaTeX
`.lytex`	LaTeX
`.lyxml`	DocBook
`.tely`	Texinfo
`.tex`	LaTeX
`.texi`	Texinfo
`.texinfo`	Texinfo
`.xml`	HTML

Lorsque le fichier source a la même extension que celle que `lilypond-book` affectera au fichier résultant et que vous lancez `lilypond-book` à partir du répertoire le contenant, vous verrez assurément un message du type « La sortie va écraser le fichier d'entrée ». Aussi ne saurions-nous trop vous conseiller d'utiliser l'option `--output`.

3.6 Modèles pour lilypond-book

Voici quelques canevas dédiés à `lilypond-book`. Si vous ne savez pas de quoi il retourne, lisez le chapitre Chapitre 3 [lilypond-book], page 22.

3.6.1 LaTeX

Vous pouvez inclure des partitions LilyPond dans un document LaTeX.

```
\documentclass[]{article}

\begin{document}

Des bananes alitées sur du LaTeX.

\begin{lilypond}
\relative {
  a'4 b c d
}
\end{lilypond}

Encore des banalités LaTeX, puis quelques options entre crochets.

\begin{lilypond}[fragment,relative=2,quote,staffsize=26,verbatim]
d4 c b a
\end{lilypond}
\end{document}
```

3.6.2 Texinfo

Un document Texinfo est tout à fait capable de comporter des fragments de partition LilyPond. Si vous ne le savez pas encore, sachez que l'intégralité de ce manuel est rédigée en Texinfo.

```
\input texinfo @node Top
@top

Du verbiage à la mode Texinfo

@lilypond
\relative {
  a4 b c d
}
@end lilypond

Toujours plus de texte Texinfo, puis des options entre crochets.

@lilypond[verbatim,fragment,ragged-right]
d4 c b a
@end lilypond

@bye
```

3.6.3 html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<!-- header_tag -->
<HTML>
<body>

<p>
Un document pour lilypond-book peut absolument mélanger musique et
texte.  Par exemple,
<lilypond>
\relative {
  a'4 b c d
}
</lilypond>
</p>

<p>
Pourquoi pas un peu plus de lilypond, avec des options pour changer :

<lilypond fragment quote staffsize=26 verbatim>
a4 b c d
</lilypond>
</p>

</body>
</html>
```

3.6.4 xelatex

```
\documentclass{article}
\usepackage{ifxetex}
\ifxetex
%pour ce qui est de xetex
\usepackage{xunicode,fontspec,xltxtra}
\setmainfont[Numbers=OldStyle]{Times New Roman}
\setsansfont{Arial}
\else
%inutile en l'absence de pdftex
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{mathptmx}%Times
\usepackage{helvet}%Helvetica
\fi
%ici les paquetages que pdftex sait interpréter
\usepackage[ngerman,finnish,english]{babel}
\usepackage{graphicx}

\begin{document}
\title{Un petit document avec LilyPond et xelatex}
\maketitle

Les commandes habituelles de \textbf{fontes} sont fonctionnelles y
compris au fil du texte, étant donné qu'\textsf{elles sont prises en
charge par \LaTeX{} and XeteX.} Lorsque vous avez besoin de commandes
particulières du style \verb+\XeTeX+, pensez à les inclure dans un
environnement \verb+\ifxetex+. Vous pourrez ainsi utiliser la \ifxetex
commande \XeTeX{} \else commande XeTeX \fi qui, elle, n'est pas reconnue
par le \LaTeX traditionnel.

Vous pouvez inclure des commandes LilyPond directement dans votre texte,
comme ici~:

\begin{lilypond}
{a2 b c'8 c' c' c'}
\end{lilypond}

\noindent
puis reprendre le fil de votre discours.

Les fontes utilisées dans les extraits LilyPond  devront être définies
au sein de l'extrait. Lisez le manuel d'utilisation si vous ne
maîtrisez pas lilypond-book.

\selectlanguage{ngerman}
Auch Umlaute funktionieren ohne die \LaTeX -Befehle, wie auch alle anderen
seltsamen Zeichen: ß, æ, ␣l, ã, č, wenn sie von der Schriftart unterstützt
werden.

\end{document}
```

3.7 Gestion de la table des matières

Les fonctions ici mentionnées sont incluses dans le paquetage OrchestralLily, disponible sur
http://repo.or.cz/w/orchestrallily.git

Certains utilisateurs privilégient la flexibilité dans la gestion du texte ; ils génèrent la table des matières à partir de LilyPond et la récupèrent dans LaTeX.

Export de la table à partir de LilyPond

Nous partons du principe que LilyPond a généré un seul fichier comportant tous les mouvement de la partition.

```
#(define (oly:create-toc-file layout pages)
  (let* ((label-table (ly:output-def-lookup layout 'label-page-table)))
    (if (not (null? label-table))
      (let* ((format-line (lambda (toc-item)
             (let* ((label (car toc-item))
                    (text  (caddr toc-item))
                    (label-page (and (list? label-table)
                                     (assoc label label-table)))
                    (page (and label-page (cdr label-page))))
               (format #f "~a, section, 1, {~a}, ~a" page text label))))
             (formatted-toc-items (map format-line (toc-items)))
             (whole-string (string-join formatted-toc-items ",\n"))
             (output-name (ly:parser-output-name))
             (outfilename (format "~a.toc" output-name))
             (outfile (open-output-file outfilename)))
        (if (output-port? outfile)
            (display whole-string outfile)
            (ly:warning (_ "Impossible d'ouvrir le fichier ~a contenant les informations de TdM") outfilename))
        (close-output-port outfile)))))

\paper {
  #(define (page-post-process layout pages) (oly:create-toc-file layout pages))
}
```

Import de la table dans LaTeX

L'entête de votre fichier LaTeX doit comporter les lignes

```
\usepackage{pdfpages}
\includescore{nomdelapartition}
```

où \includescore est défini ainsi :

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% \includescore{PossibleExtension}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Read in the TOC entries for a PDF file from the corresponding .toc file.
% This requires some heave latex tweaking, since reading in things from a file
% and inserting it into the arguments of a macro is not (easily) possible

% Solution by Patrick Fimml on #latex on April 18, 2009:
% \readfile{filename}{\variable}
% reads in the contents of the file into \variable (undefined if file
% doesn't exist)
\newread\readfile@f
\def\readfile@line#1{%
{\catcode`\^^M=10\global\read\readfile@f to \readfile@tmp}%
\edef\do{\noexpand\g@addto@macro{\noexpand#1}{\readfile@tmp}}\do%
\ifeof\readfile@f\else%
\readfile@line{#1}%
\fi%
}
\def\readfile#1#2{%
```

```
\openin\readfile@f=#1 %
\ifeof\readfile@f%
\typeout{No TOC file #1 available!}%
\else%
\gdef#2{}%
\readfile@line{#2}%
\fi
\closein\readfile@f%
}%


\newcommand{\includescore}[1]{
\def\oly@fname{\oly@basename\@ifmtarg{#1}{}{_#1}}
\let\oly@addtotoc\undefined
\readfile{\oly@xxxxxxxxx}{\oly@addtotoc}
\ifx\oly@addtotoc\undefined
\includepdf[pages=-]{\oly@fname}
\else
\edef\includeit{\noexpand\includepdf[pages=-,addtotoc={\oly@addtotoc}]
{\oly@fname}}\includeit
\fi
}
```

3.8 Autres méthodes d'association texte-musique

D'autres moyens de mélanger musique et texte sans recourir à `lilypond-book` sont abordés au chapitre Section 4.4 [Inclusion de partition LilyPond dans d'autres programmes], page 49.

4 Programmes externes

LilyPond peut interagir avec d'autres programmes, selon différentes manières.

4.1 Pointer-cliquer

Le pointer-cliquer (*point and click*) permet de se retrouver directement dans le fichier source, à la note que l'on pointe dans le visionneur de PDF. Ceci facilite grandement le repérage des erreurs à partir du fichier imprimable.

4.1.1 Configuration du système

Lorsque cette fonctionnalité est active, LilyPond ajoute des hyperliens au fichier PDF ou SVG. Ces liens sont transmis à un « URI helper » ou au navigateur internet, qui se charge d'ouvrir un éditeur de texte à l'endroit même où le curseur pointe.

Afin que cette chaîne de traitement soit pleinement opérationnelle, il faut configurer votre visionneur de PDF de façon à ce qu'il suive les liens grâce au script `lilypond-invoke-editor` fourni avec LilyPond.

`lilypond-invoke-editor` est un petit programme assistant. Il se charge d'appeler un éditeur pour les identifiants de ressource (*URI*) de type `textedit`, et un navigateur pour les autres. Il teste en outre la variable d'environnement `EDITOR` pour les cas suivant :

`emacs` sera alors lancée la commande

`emacsclient --no-wait +`*`line`*`:`*`column file`*

`gvim` sera alors lancée la commande

`gvim --remote +:`*`line`*`:normcolumn`*` file`*

`nedit` sera alors lancée la commande

`nc -noask +`*`line file`*`'`

La variable d'environnement `LYEDITOR` permet d'anticiper cette affectation, puisqu'elle contient la commande qui permet de lancer l'éditeur en tenant compte des valeurs respectives de `%(file)s`, `%(column)s` et`%(line)s`. Par exemple,

```
emacsclient --no-wait +%(line)s:%(column)s %(file)s
```

en variable d'environnement `LYEDITOR` revient au lancement d'un client emacs standard.

Utilisation avec Xpdf

Pour Xpdf, sous UNIX, vous devrez insérer la ligne suivante dans le fichier `xpdfrc` – soit `/etc/xpdfrc`, soit dans votre répertoire personnel `$HOME/.xpdfrc`.

```
urlCommand     "lilypond-invoke-editor %s"
```

Dans un environnement Ubuntu, il est fort probable que la version de Xpdf installée avec le système plante à l'ouverture de tout fichier PDF – c'est un problème connu et persistant depuis plusieurs années et dû à des incohérences de bibliothèques. La solution consiste alors à installer une version à jour des paquetages '`xpdf`' et '`libpoppler`' directement à partir de Debian. Une fois assuré que tout fonctionne correctement, la commande

```
sudo apt-mark hold xpdf
```

permet d'empêcher Ubuntu de le remplacer par un paquetage défectueux à la prochaine « mise à jour ».

Utilisation avec GNOME 2

En ce qui concerne l'environnement GNOME 2 et les lecteur de PDF associés, la succession de commandes suivante permet de régler le système pour la gestion des URI par 'textedit:'

```
gconftool-2 -t string -s /desktop/gnome/url-handlers/textedit/command "lilypond-invoke-editor %s"
gconftool-2 -s /desktop/gnome/url-handlers/textedit/needs_terminal false -t bool
gconftool-2 -t bool -s /desktop/gnome/url-handlers/textedit/enabled true
```

Après ces invocations,

```
gnome-open textedit:///etc/issue:1:0:0
```

devrait appeler `lilypond-invoke-editor` pour ouvrir les fichiers.

Utilisation avec GNOME 3

En ce qui concerne l'environnement GNOME 3, les URI sont gérés par la surcouche « gvfs » au lieu de « gconf ». Il faut donc créer un fichier dans un répertoire local tel que `/tmp`, que l'on appelera `lilypond-invoke-editor.desktop`. Il devra avoir le contenu suivant :

```
[Desktop Entry]
Version=1.0
Name=lilypond-invoke-editor
GenericName=Textedit URI handler
Comment=URI handler for textedit:
Exec=lilypond-invoke-editor %u
Terminal=false
Type=Application
MimeType=x-scheme-handler/textedit;
Categories=Editor
NoDisplay=true
```

puis exécuter les commandes

```
xdg-desktop-menu install ./lilypond-invoke-editor.desktop
xdg-mime default lilypond-invoke-editor.desktop x-scheme-handler/textedit
```

Après cette invocation,

```
gnome-open textedit:///etc/issue:1:0:0
```

devrait appeler `lilypond-invoke-editor` pour ouvrir les fichiers.

Configuration spécifique à Evince

Il se peut que, bien que `gnome-open` soit fonctionnel, Evince refuse d'ouvir les liens pointer-cliquer pour des raisons d'autorisation. S'impose alors une modification du profil `Apparmor` d'Evince ; c'est lui qui contrôle le type d'action qu'Evince est autorisé à réaliser.

Sur une distribution Ubuntu, cela consiste à éditer le fichier `/etc/apparmor.d/local/usr.bin.evince` et lui ajouter les lignes suivantes :

```
# Pour les liens Textedit
/usr/local/bin/lilypond-invoke-editor Cx -> sanitized_helper,
```

puis lancer la commande

```
sudo apparmor_parser -r -T -W /etc/apparmor.d/usr.bin.evince
```

Evince devrait alors être en mesure d'ouvrir les liens pointer-cliquer. Une telle configuration devrait être fonctionnelle pour d'autres visionneurs.

Activation du pointer-cliquer

La fonctionnalité de « pointer-cliquer » est activée par défaut pour ce qui est des fichiers PDF et SVG.

L'option pointer-cliquer accroît la taille des fichiers de manière significative. Afin de réduire la taille de ces fichiers (ainsi que du PS), il est toujours possible de désactiver le pointer-cliquer en ajoutant

```
\pointAndClickOff
```

dans le fichier `.ly`. Il peut alors être activé de manière explicite grâce à

```
\pointAndClickOn
```

Le pointer-cliquer peut aussi être désactivé au moment de la compilation en ligne de commande :

```
lilypond -dno-point-and-click file.ly
```

Note : Lorsqu'un fichier LilyPond est destiné à être redistribué, pensez à désactiver le pointer-cliquer, de telle sorte que les chemins d'accès et autres informations propres à votre système ne se retrouvent pas inclus dans le fichier PDF.

Pointer-cliquer sélectif

Pour certaines applications interactives, il est parfois préférable de limiter la fonctionnalité du pointer-cliquer à quelques éléments seulement. Par exemple, si vous avez l'intention de créer une application lançant l'audio et la vidéo à partir d'une note en particulier, il serait mal venu qu'un clic sur la note vous amène à l'altération ou une liaison qui l'affecterait.

Les événements générateurs de lien peuvent se gérer :

- En dur dans votre fichier `.ly` :

  ```
  \pointAndClickTypes #'note-event
  \relative {
    c'2\f( f)
  }
  ```

  ou

  ```
  #(ly:set-option 'point-and-click 'note-event)
  \relative {
    c'2\f( f)
  }
  ```

- En ligne de commande :

  ```
  lilypond -dpoint-and-click=note-event exemple.ly
  ```

Plusieurs types d'événement peuvent être mentionnés :

- En dur dans votre fichier `.ly` :

  ```
  \pointAndClickTypes #'(note-event dynamic-event)
  \relative {
    c'2\f( f)
  }
  ```

  ou

  ```
  #(ly:set-option 'point-and-click '(note-event dynamic-event))
  \relative {
    c'2\f( f)
  }
  ```

- En ligne de commande :

  ```
  lilypond \
    -e"(ly:set-option 'point-and-click '(note-event dynamic-event))" \
    exemple.ly
  ```

4.2 LilyPond et les éditeurs de texte

Certains éditeurs de texte prennent en charge LilyPond.

Mode Emacs

Emacs dispose d'un `lilypond-mode` qui fournit l'autocomplétion des mots-clés, l'indentation, les appariements spécifiques à LilyPond, la coloration syntaxique, ainsi que des raccourcis pour compiler et consulter les manuels de LilyPond en mode info. Si le `lilypond-mode` n'était pas installé sur votre système, procédez comme ci-dessous.

Le répertoire `elisp` inclus dans les sources contient aussi un mode pour saisir la musique et lancer LilyPond. Faites `make install` pour l'installer dans votre *elispdir*. Le fichier `lilypond-init.el` devrait trouver sa place dans *load-path*`/site-start.d/` ou bien ajouté à votre `~/.emacs` ou `~/.emacs.el`.

En tant que simple utilisateur, vous pouvez toujours ajouter votre propre répertoire (par exemple `~/site-lisp/`) à votre *load-path* en ajoutant la ligne suivante – modifiée en conséquence – à votre `~/.emacs` :

```
(setq load-path (append (list (expand-file-name "~/site-lisp")) load-path))
```

Mode Vim

En ce qui concerne Vim (`http://www.vim.org`), LilyPond fournit tout le nécessaire pour gérer la coloration syntaxique et l'indentation. Le mode spécifique à Vim doit être indiqué dans le fichier `$HOME/.vimrc`. Localisez ce fichier, ou créez-le, et ajoutez-y les trois lignes suivantes :

```
filetype off
set runtimepath+=/usr/local/share/lilypond/current/vim/
filetype on
syntax on
```

Si LilyPond est installé dans un autre répertoire que `/usr/local/`, modifiez ce chemin en conséquence. Pour de plus amples détails, consultez Section "Autres sources de documentation" dans *Manuel d'initiation*.

Autres éditeurs

LilyPond est pris en charge par d'autres éditeurs, aussi bien en mode texte qu'avec une interface graphique. Dans la mesure où leurs fichiers de configuration ne sont pas fournis avec LilyPond, nous vous invitons à consulter leur documentation pour plus d'information. Certains de ces éditeurs sont mentionnés à la page Section "Facilités d'édition" dans *Informations générales*.

4.3 Conversion à partir d'autres formats

La musique peut aussi être récupérée par importation d'un autre format. Ce chapitre passe en revue les différents outils prévus à cet effet et inclus dans la distribution. Il existe d'autres outils qui permettent de générer du code LilyPond, comme par exemple des séquenceurs en mode graphique ou des convertisseurs XML. Pour plus de détails, rendez-vous sur le site (`http://lilypond.org`).

Il s'agit de programmes distincts de `lilypond` qui se lancent en ligne de commande. Pour plus de précisions, reportez-vous au chapitre Section 1.2 [Utilisation en ligne de commande], page 1. Si vous utilisez MacOS 10.3 ou 10.4 et rencontrez quelque problème avec l'un de ces scripts, comme `convert-ly`, reportez-vous à la page Section "MacOS X" dans *Informations générales*.

Problèmes connus et avertissements

Les développeurs ne sont malheureusement pas suffisamment nombreux et disponibles pour maintenir à jour ces programmes, considérez-les donc *en l'état*. Nous acceptons les patches avec

plaisir, mais il y a peu de chance pour que nous soyons en mesure de résoudre les bogues de ces programmes.

4.3.1 Utilisation de `midi2ly`

`midi2ly` traduit un fichier MIDI de Type 1 en un fichier source LilyPond.

MIDI (Music Instrument Digital Interface) constitue un standard pour les instruments. Il spécifie le câblage, un protocole série et un format de fichier. Le format de fichier MIDI est de ce fait un format standard pour exporter de la musique à partir d'autres programmes, et cette faculté prend tout son intérêt lorsqu'un programme dispose d'un convertisseur capable d'importer directement un fichier.

`midi2ly` convertit les pistes en contextes de portée (Section "Staff" dans *Référence des propriétés internes*) et les canaux en contextes de voix (Section "Voice" dans *Référence des propriétés internes*). Les hauteurs sont rendues en mode relatif, et les durées spécifiées lorsque nécessaire.

Vous pouvez enregistrer un fichier MIDI grâce à un clavier électronique et ensuite le convertir en fichier `.ly`. Néanmoins le rythme humain n'a pas une précision suffisante pour qu'une conversion MIDI à ly ne se fasse sans surprise. En le couplant à une quantisation (options `-s` et `-d`), `midi2ly` tente de compenser dans la mesure du possible ces problèmes de temporisation. C'est la raison pour laquelle le recours à `midi2ly` n'est pas recommandé pour des fichiers midi générés manuellement.

Pour le lancer en ligne de commande, procédez ainsi :

```
midi2ly [option]... fichier-midi
```

Notez bien que, par « ligne de commande », nous parlons de la ligne de commande du système. Pour plus de précisions, reportez-vous à Section 4.3 [Conversion à partir d'autres formats], page 45.

`midi2ly` accepte les options suivantes :

`-a, --absolute-pitches`
: Rendu en hauteurs absolues.

`-d, --duration-quant=`*`DUR`*
: Quantiser les durées à partir de *DUR*.

`-e, --explicit-durations`
: Rendu explicite des durées.

`-h, --help`
: Afficher un résumé des utilisations.

`-k, --key=`*`acc`*`[:`*`minor`*`]`
: Déterminer la tonalité par défaut. $acc > 0$ fixe le nombre de dièses, $acc < 0$ le nombre de bémols. Une tonalité mineure est spécifiée par l'emploi de `:1`.

`-o, --output=`*`file`*
: Générer le résultat dans le fichier *file*.

`-s, --start-quant=`*`DUR`*
: Quantiser le début des notes sur *DUR*.

`-t, --allow-tuplet=`*`DUR*NUM/DEN`*
: Accepter des n-olets de valeur *DUR**NUM/DEN*.

`-v, --verbose`
: Mode verbeux.

`-V, --version`
Afficher le numéro de version.

`-w, --warranty`
Afficher les mentions de garantie et de copyright.

`-x, --text-lyrics`
Interpréter le texte comme des paroles.

Problèmes connus et avertissements

Le tuilage en arpège ne sera pas rendu correctement. La première note sera lue et les suivantes ignorées. Affectez-leur une durée unique et ajoutez une indication de phrasé ou de pédale.

4.3.2 Utilisation de `musicxml2ly`

MusicXML (`http://www.musicxml.org/`) est un dialecte XML pour représenter la notation musicale.

`musicxml2ly` extrait, à partir d'un fichier MusicXML, les notes, articulations, structure de la partition, paroles, etc. et les écrit dans un fichier `.ly`. Il se lance ainsi en ligne de commande :

`musicxml2ly [`*`option`*`]... `*`fichier-xml`*

Notez bien que, par « ligne de commande », nous parlons de la ligne de commande du système. Pour plus de précisions, reportez-vous à Section 4.3 [Conversion à partir d'autres formats], page 45.

Si le nom du fichier est `-`, `musicxml2ly` lira directement à partir de la ligne de commande.

`musicxml2ly` accepte les options suivantes :

`-a, --absolute`
Rendu en hauteurs absolues.

`-h, --help`
Afficher un résumé des utilisations.

`-l, --language=LANG`
Utiliser une autre définition linguistique (LANG), comme par exemple *deutsch* pour des noms de notes en allemand.

`--loglevel=`*`loglevel`*
Détermine le degré de verbosité à *loglevel*. Les valeurs autorisées sont `NONE`, `ERROR`, `WARNING`, `PROGRESS` (par défaut) et `DEBUG`.

`--lxml`
Utiliser le paquetage Python `lxml.etree`, moins gourmand en mémoire et temps de calcul, pour effectuer l'analyse XML.

`-m, --midi`
Ajouter un bloc `\midi`.

`--nd, --no-articulation-directions`
Ne pas convertir la direction (`^`, `_` ou `-`) des articulations, nuances, etc.

`--no-beaming`
Ne pas convertir les informations de ligature ; laisser LilyPond gérer les ligatures automatiquement.

`-o, --output=`*`file`*
Générer le résultat dans le fichier *fichier*. S'il n'est pas déterminé, ce sera *fichier-xml*`.ly` ; `-` produira le résultat sur la sortie standard (*stdout*).

`-r, --relative`
Rendu en hauteurs relatives (mode par défaut).

`-v, --verbose`
: Mode verbeux.

`--version`
: Afficher le numéro de version.

`-z, --compressed`
: Le fichier d'entrée est un fichier MusicXML zippé.

4.3.3 Utilisation de `abc2ly`

Note : Ce programme ne bénéficie d'aucune maintenance. Il est susceptible d'être supprimé des versions futures de LilyPond.

ABC est un format relativement simple basé sur l'ASCII. Sa description est disponible sur le site d'ABC (`http://www.walshaw.plus.com/abc/learn.html`).

`abc2ly` traduit du format ABC au format LilyPond. Pour le lancer en ligne de commande, procédez ainsi :

```
abc2ly [option]... fichier-abc
```

`abc2ly` accepte les options suivantes :

`-b, --beams=None`
: Préserver la notion de lien de croches propre à ABC.

`-h, --help`
: Afficher un résumé des utilisations.

`-o, --output=file`
: Générer le résultat dans le fichier *file*.

`-s, --strict`
: Être strict sur la réussite.

`--version`
: Afficher le numéro de version.

Il est possible d'ajouter des bribes de code LilyPond dans un fichier source ABC. Ainsi, l'assertion

```
%%LY voices \set autoBeaming = ##f
```

aura pour conséquence d'insérer le texte qui suit le mot-clé « voices » dans la voix correspondante du fichier LilyPond.

De la même manière,

```
%%LY slyrics more words
```

placera le texte suivant le mot-clé « slyrics » dans une ligne de paroles.

Problèmes connus et avertissements

Le standard ABC n'est pas si « standard » que cela. Pour des fonctionnalités étendues, comme la polyphonie, existent différentes conventions.

Un fichier comportant plusieurs morceaux ne peut être converti.

ABC synchronise paroles et musique en début de ligne ; `abc2ly` ne le fait pas.

`abc2ly` ignore les ligatures ABC.

4.3.4 Utilisation de `etf2ly`

> **Note :** Ce programme ne bénéficie d'aucune maintenance. Il est susceptible d'être supprimé des versions futures de LilyPond.

ETF (Enigma Transport Format) est l'un des formats utilisés par le logiciel Finale, édité par Coda Music Technology. `etf2ly` convertit partiellement les fichiers ETF en fichiers source LilyPond.

Pour le lancer en ligne de commande, procédez ainsi :

`etf2ly [`*`option`*`]... `*`fichier-etf`*

Notez bien que, par « ligne de commande », nous parlons de la ligne de commande du système. Pour plus de précisions, reportez-vous à Section 4.3 [Conversion à partir d'autres formats], page 45.

`etf2ly` accepte les options suivantes :

`-h, --help`
: Afficher cette aide.

`-o, --output=`*`file`*
: Générer le résultat dans le fichier *file*.

`--version`
: Afficher le numéro de version.

Problèmes connus et avertissements

La liste des scripts d'articulation est incomplète. Les mesures vides perturbent `etf2ly`. Les séquences de notes d'ornement ne se terminent pas de manière satisfaisante.

4.3.5 Autres formats

LilyPond ne prend pas en charge d'autre format. Cependant, certains outils indépendants permettent de générer des fichiers LilyPond, comme indiqué à la page Section "Facilités d'édition" dans *Informations générales*.

4.4 Inclusion de partition LilyPond dans d'autres programmes

Nous allons nous intéresser ici à différents moyens d'associer texte et musique, en laissant de côté l'automatisation grâce à `lilypond-book`.

4.4.1 LuaTex

Tout comme avec `lilypond-book`, les utilisateurs de LuaTex peuvent intégrer le résultat de LilyPond dans leurs documents à l'aide de l'extension lyluatex (`https://github.com/jperon/lyluatex/blob/master/README.md`).

OpenOffice et LibreOffice

OOoLilyPond (`http://ooolilypond.sourceforge.net`) permet d'insérer directement des partitions LilyPond dans OpenOffice ou LibreOffice. Bien que son développement soit suspendu, cette extension est opérationnelle avec la version 4 des deux suites.

Autres programmes

Pour les nombreux programmes qui sont capables de contenir des images, que ce soit au format `PNG`, `EPS` ou `PDF`, mieux vaut utiliser `lilypond` que `lilypond-book`. Chaque extrait devra être

généré séparément avant d'être inséré dans votre document. Consultez la documentation du logiciel en question quant à l'insertion de fichiers provenant d'autres sources.

Les options suivantes vous permettront de réduire notablement les contours de l'image LilyPond :

```
\paper{
  indent=0\mm
  line-width=120\mm
  oddFooterMarkup=##f
  oddHeaderMarkup=##f
  bookTitleMarkup = ##f
  scoreTitleMarkup = ##f
}

... musique ...
```

En procédant comme ci-après, vous obtiendrez des images `EPS` :

```
lilypond -dbackend=eps -dno-gs-load-fonts -dinclude-eps-fonts monfichier.ly
```

des images `PNG` :

```
lilypond -dbackend=eps -dno-gs-load-fonts -dinclude-eps-fonts --png monfichier.ly
```

ou bien des images `PNG` avec transparence :

```
lilypond -dbackend=eps -dno-gs-load-fonts -dinclude-eps-fonts -dpixmap-format=pngalpha --png myfile.ly
```

Si d'aventure vous deviez recopier un certain nombre d'extraits d'une même partition, vous pouvez recourir à « l'emporte pièce » – la fonction *clip systems* – comme indiqué au chapitre Section "Extraction de fragments musicaux" dans *Manuel de notation*.

4.5 Inclusion du travail des autres

Certains utilisateurs ont, pour obtenir des effets particuliers, écrit des fichiers qu'il est toujours possible d'intégrer à LilyPond avec l'instruction `\include`. Plusieurs sont même désormais inclus dans le programme. Voir aussi Section "Travail sur des fichiers texte" dans *Manuel de notation*.

4.5.1 MIDI et articulations

Le projet *articulate* (`http://www.nicta.com.au/people/chubbp/articulate`) (site en anglais) s'est donné pour objectif d'améliorer la sortie MIDI de LilyPond. Les notes qui ne sont pas liées sont ainsi raccourcies dans le but « d'articuler ». Ce raccourcissement dépend de l'articulation appliquée à la note : un *staccato* raccourcira la note de moitié, un *tenuto* lui gardera sa durée entière. . . Voir Section "Amélioration du rendu MIDI" dans *Manuel de notation*.

5 Suggestions pour la saisie de fichiers LilyPond

Maintenant vous êtes prêt à travailler sur de plus gros fichiers LilyPond – des pièces entières, et plus seulement les petits exemples du tutoriel. Mais comment devriez-vous vous y prendre ?

Tant que LilyPond parvient à comprendre vos fichiers et produit le résultat que vous souhaitez, peu importe la manière dont le code est organisé. Néanmoins, quelques critères doivent être pris en compte lorsque l'on écrit un fichier LilyPond.

- Si vous faites une erreur, la structure même du fichier LilyPond peut permettre de la localiser plus ou moins facilement.
- Et si vous souhaitez partager vos fichiers avec quelqu'un d'autre, ou si vous souhaitez modifier vos propres fichiers dans quelques années ? Si certains fichiers LilyPond sont compréhensibles au premier coup d'œil, d'autres vous feront vous arracher les cheveux pendant une heure.
- Et si vous souhaitez mettre à jour votre fichier pour l'utiliser avec une version plus récente de LilyPond ? La syntaxe du langage d'entrée change parfois lorsque LilyPond s'améliore. La plupart des changements peuvent être appliqués automatiquement avec `convert-ly`, mais quelques-uns peuvent requérir une intervention manuelle. Vos fichiers LilyPond peuvent être structurés de manière à faciliter leur mise à jour.

5.1 Suggestions générales

Voici quelques conseils qui vous permettront d'éviter, voire même résoudre, la plupart des problèmes de saisie.

- **Ajoutez toujours le numéro de version dans chaque fichier**, quelle que soit sa taille. Par expérience, il est très difficile de se rappeler quelle version de LilyPond a servi au moment de la création d'un fichier. Ceci s'avèrera d'autant plus utile lors d'une Section "mise à jour" dans *Utilisation des programmes* (`convert-ly` requiert la présence d'une ligne `\version`) ou si vous transmettez votre fichier à d'autres utilisateurs – y compris pour demander de l'aide sur les listes de diffusion. Vous noterez par ailleurs que tous les fichiers modèles de LilyPond ont une mention `\version`.
- **Une mesure par ligne de texte**. Si la musique en elle-même ou le résultat que vous désirez contient quelque chose de compliqué, il est souvent bon de n'écrire qu'une seule mesure par ligne. Économiser de la place en tassant huit mesures par ligne, ça ne vaut pas vraiment le coup si l'on doit corriger vos fichiers.
- **Ajoutez des contrôles Section "d'octaviation" dans** *Manuel de notation* **et Section "de limite ou numéro de mesure" dans** *Manuel de notation*. Si vous avez ajouté des contrôles de loin en loin et que vous faites une erreur, vous pourrez la retrouver plus rapidement. « De loin en loin », qu'est-ce à dire ? Cela dépend de la complexité de la musique. Pour de la musique très simple, à certains endroits stratégiques. Pour de la musique très complexe ou avec une multiplicité de voix, peut-être à chaque mesure.
- **Ajoutez des commentaires**. Utilisez soit des numéros de mesure (assez souvent), soit des références au contenu musical – « second thème des violons », « quatrième variation », etc. Vous pouvez ne pas avoir besoin des commentaires lorsque vous écrivez une pièce pour la première fois, mais si vous souhaitez y revenir deux ou trois ans plus tard pour changer quelque chose, ou si vous donnez le fichier source à un ami, ce sera beaucoup plus difficile de déterminer vos intentions ou la manière dont votre fichier est structuré si vous n'y avez pas adjoint de commentaires.
- **Mentionnez les durées au début de chaque section ou variable**. Si vous saisissez `c4 d e` au début d'une phrase, vous vous épargnerez des problèmes si, plus tard, vous modifiez votre musique.

- **Indentez les accolades et indications de musique en parallèle**. Beaucoup de problèmes viennent d'un défaut de parité entre `{` et `}` ou `<<` et `>>`. Par exemple,

```
\new Staff {
  \relative {
    r4 g'8 g c8 c4 d |
    e4 r8 |
    % Ossia section
    <<
      { f8 c c | }
      \new Staff {
        f8 f c |
      }
    >>
    r4 |
  }
}
```

  est bien plus facile à appréhender que

```
\new Staff { \relative { r4 g'8 g c4 c8 d | e4 r8
% Ossia section
<< { f8 c c } \new Staff { f8 f c } >> r4 | } }
```

- **Séparez les affinages de mise en forme** de la musique elle-même, ne serait-ce qu'en positionnant les dérogations au sein du bloc `\layout` :

```
\score {
  ...musique...
  \layout {
   \override TabStaff.Stemstencil = ##f
 }
}
```

  Ceci n'aura par pour effet de générer un contexte supplémentaire, mais s'appliquera dès sa création. Voyez Section "Économie de saisie grâce aux identificateurs et fonctions" dans *Manuel d'initiation* et Section "Feuilles de style" dans *Manuel d'initiation*.

5.2 Gravure de musique existante

Si vous saisissez de la musique à partir d'une partition existante, c'est-à-dire de la musique déjà écrite,

- n'entrez qu'un seul système de la partition originale à la fois – avec toujours une seule mesure par ligne de texte –, et vérifiez chaque système lorsqu'il est terminé. Vous pouvez utiliser les commandes `showLastLength` et `showFirstLength` pour accélérer la compilation – voir Section "Ignorer des passages de la partition" dans *Manuel de notation* ;
- définissez `mBreak = { \break }` et insérez `\mBreak` dans le fichier d'entrée pour obtenir des sauts de ligne identiques à la partition originale. Cela facilite la comparaison entre la partition originale et la partition de LilyPond. Lorsque vous avez fini de relire votre musique, vous pouvez définir `mBreak = { }` pour enlever tous ces sauts de ligne, et laisser LilyPond placer les sauts de ligne selon son propre algorithme ;
- encadrez les notes d'une partie pour instrument transpositeur dans un

```
\transpose c tonalité-naturelle {...}
```

  (où *tonalité-naturelle* correspond à celle de l'instrument en question) de telle sorte que la musique comprise dans cette variable se retrouve en ut. Vous pourrez toujours transposer à l'inverse si besoin lorsque vous ferez appel à cette variable. Des erreurs de transposition

seront moins susceptibles de se produire si la musique de toutes les variables est dans la même et unique tonalité.

De la même manière, prenez toujours le do comme note de départ ou d'arrivée. Ceci aura pour simple conséquence que les autres tonalités que vous utiliserez seront celles propres à chacun des instruments – sib pour une trompette en si bémol, ou lab pour une clarinette en la bémol.

5.3 Projets d'envergure

Lorsque l'on travaille sur un gros projet, il devient vital de structurer clairement ses fichiers LilyPond.

- **Utilisez un identificateur pour chaque voix**, avec un minimum de structure dans la définition. La structure de la section `\score` est la plus susceptible de changer, notamment dans une nouvelle version de LilyPond, alors que la définition du `violon` l'est beaucoup moins.

```
violon = \relative {
g'4 c'8. e16
}
...
\score {
 \new GrandStaff {
   \new Staff {
     \violon
   }
 }
}
```

- **Séparez les retouches des définitions de musique**. Nous vous avons déjà invité à adopter une telle pratique, qui par ailleurs devient vitale pour des projets d'importance. Nous pouvons avoir besoin de changer la définition de `fpuisp`, mais dans ce cas nous n'aurons besoin de le faire qu'une seule fois, et nous pourrons encore éviter de modifier quoi que ce soit à l'intérieur de la définition du `violon`.

```
fpuisp = _\markup{
 \dynamic f \italic \small { 2nd } \hspace #0.1 \dynamic p }
violon = \relative {
g'4\fpuisp c'8. e16
}
```

5.4 Résolution de problèmes

Tôt ou tard, vous écrirez un fichier que LilyPond ne peut pas compiler. Les messages que LilyPond affiche peuvent vous aider à trouver l'erreur, mais dans beaucoup de cas vous aurez besoin de faire quelques recherches pour déterminer la source du problème.

Pour ce faire, les outils les plus puissants sont le commentaire de fin de ligne, indiqué par `%`, et le commentaire multilignes (ou bloc de commentaire), indiqué par `%{ ... %}`. Si vous ne pouvez localiser le problème, commencez par mettre en commentaire de grandes parties de votre fichier source. Après avoir mis en commentaire une section, essayez de compiler à nouveau. Si cela fonctionne, c'est que le problème se situe dans cette partie du fichier. Si cela ne fonctionne pas, continuez à mettre en commentaire d'autres sections, jusqu'à ce que vous ayez quelque chose qui compile.

Dans un cas extrême, vous pourriez en arriver à

```
\score {
  <<
```

```
    % \melodie
    % \harmonie
    % \basse
  >>
  \layout{}
}
```

c'est-à-dire un fichier sans aucune musique.

Si cela se produit, ne vous découragez pas. Décommentez un peu, la partie de basse par exemple, et voyez si ça fonctionne. Si ce n'est pas le cas, placez en commentaire toute la partie de basse, mais laissez `\basse` décommenté dans le bloc `\score`.

```
basse = \relative {
%{
  c'4 c c c
  d d d d
%}
}
```

Maintenant commencez à décommenter petit à petit la partie de `basse` jusqu'à ce que vous localisiez la ligne qui pose problème.

Une autre technique de débogage très utile est la construction d'un Section "exemple minimaliste" dans *Informations générales*.

5.5 De la commande make et des fichiers Makefile

La plupart des plates-formes sur lesquelles tourne LilyPond disposent d'un logiciel appelé `make`. Ce logiciel va lire un fichier spécial, nommé `Makefile`, qui contient tout ce qu'il faut – les dépendances entre certains fichiers, les instructions successives à traiter par le système – pour aboutir au fichier que vous désirez obtenir. Il pourrait par exemple contenir tout ce qu'il faut pour produire `ballade.pdf` et `ballade.midi` à partir de `ballade.ly` en lançant LilyPond.

La création d'un `Makefile` peut se révéler pertinente pour certains projets, que ce soit par simple goût personnel ou bien par respect de ceux qui pourront accéder à vos sources. Cette manière de procéder est particulièrement indiquée lorsque vous travaillez sur un projet de grande envergure impliquant de nombreuses inclusions de fichiers et différentes éditions – par exemple un conducteur et un matériel d'orchestre complet avec la partition pour le chef et une partition séparée pour chacun des pupitres – ou bien si votre projet requiert certaines commandes particulières comme `lilypond-book`. Les *Makefiles* varient tant en complexité qu'en flexibilité selon les besoin et les aptitudes de celui qui les crée. Le programme GNU Make est installé par défaut sur les distributions GNU/Linux et sur MacOS X, et il en existe une version pour les environnements Windows.

Consultez le **GNU Make Manual** pour plus de détails sur ce dont `make` est capable – vous pourrez même en trouver des versions françaises à l'aide des moteurs de recherche –, dans la mesure où ce qui suit ne donne qu'un bref aperçu de ses possibilités.

Les commandes permettant de définir les règles diffèrent selon la plate-forme : si les différents GNU/Linux et MacOS X utilisent `bash`, Windows utilise `cmd`. Dans le cas de MacOS X, vous devrez toutefois configurer votre système de telle sorte qu'il utilise l'interpréteur en ligne de commande. Voici quelques exemples de fichier *Makefile*, avec une version pour GNU/Linux ou MacOS et une pour Windows.

Pour commencer, une pièce à quatre mouvements pour orchestre et dont les fichiers sont répartis selon l'arborescence suivante :

```
Symphonie/
|-- MIDI/
```

```
|-- Makefile
|-- Notes/
|   |-- alto.ily
|   |-- cor.ily
|   |-- cello.ily
|   |-- figures.ily
|   |-- hautbois.ily
|   |-- trioCordes.ily
|   |-- violonUn.ily
|   `-- violonDeux.ily
|-- Partitions/
|   |-- symphonie.ly
|   |-- symphonieI.ly
|   |-- symphonieII.ly
|   |-- symphonieIII.ly
|   `-- symphonieIV.ly
|-- PDF/
|-- Pupitres/
|   |-- symphonie-alto.ly
|   |-- symphonie-cello.ly
|   |-- symphonie-cor.ly
|   |-- symphonie-hautbois.ly
|   |-- symphonie-violonUn.ly
|   `-- symphonie-violonDeux.ly
`-- symphonieDefs.ily
```

Les fichiers `.ly` des répertoires `Partitions` et `Pupitres` récupéreront la notation des fichiers `.ily` contenus dans le répertoire `Notes` :

```
%%% début du fichier "symphonie-cello.ly"
\include ../symphonieDefs.ily
\include ../Notes/cello.ily
```

Le *Makefile* répertorie des cibles correspondant à `score` (l'intégrale au format conducteur), `mouvements` (chacun des mouvements au format conducteur) et `pupitres` (une partition par pupitre). Il contient aussi une cible `archive` chargée de générer une archive des fichiers source qui pourra être diffusée sur la toile ou transmise par courriel. Voici ce que contiendrait ce *Makefile* pour GNU/Linux ou MacOS X. Ce fichier doit être enregistré sous le nom de `Makefile` à la racine du projet – ici `Symphonie`.

Note : Lorsque vous définissez une cible ou une règle sur plusieurs lignes, les lignes à partir de la deuxième **doivent** débuter par une tabulation, non pas par des espaces.

```
# Le préfixe au nom des fichiers résultants
piece = symphonie
# Détermination du nombre de processeurs
CPU_CORES=`cat /proc/cpuinfo | grep -m1 "cpu cores" | sed s/".*: "//`
# La commande d'appel à lilypond
LILY_CMD = lilypond -ddelete-intermediate-files \
                    -dno-point-and-click -djob-count=$(CPU_CORES)

# Les suffixes utilisés dans ce Makefile
.SUFFIXES: .ly .ily .pdf .midi
```

```
# Les fichiers sources et résultants sont recherchés dans les répertoires
# listés dans la variable VPATH.  Ceux-ci sont tous des sous-répertoires
# du répertoire courant (fourni par la variable de GNU make `CURDIR').
VPATH = \
  $(CURDIR)/Partitions \
  $(CURDIR)/PDF \
  $(CURDIR)/Pupitres \
  $(CURDIR)/Notes

# La règle type pour créer un PDF et un MIDI à partir d'un fichier
# source LY.
# Les .pdf résultants iront dans le sous-répertoire "PDF" et les fichiers
# .midi dans le sous-répertoire "MIDI".
%.pdf %.midi: %.ly
        $(LILY_CMD) $<; \       # cette ligne commence par une tabulation
        if test -f "$*.pdf"; then \
            mv "$*.pdf" PDF/; \
        fi; \
        if test -f "$*.midi"; then \
            mv "$*.midi" MIDI/; \
        fi

notes = \
  alto.ily \
  cello.ily \
  cor.ily \
  hautbois.ily \
  violonUn.ily \
  violonDeux.ily

# Les dépendances selon le mouvement.
$(piece)I.pdf: $(piece)I.ly $(notes)
$(piece)II.pdf: $(piece)II.ly $(notes)
$(piece)III.pdf: $(piece)III.ly $(notes)
$(piece)IV.pdf: $(piece)IV.ly $(notes)

# Les dépendances pour la partition intégrale.
$(piece).pdf: $(piece).ly $(notes)

# Les dépendances pour les pupitres.
$(piece)-alto.pdf: $(piece)-alto.ly alto.ily
$(piece)-cello.pdf: $(piece)-cello.ly cello.ily
$(piece)-cor.pdf: $(piece)-cor.ly cor.ily
$(piece)-hautbois.pdf: $(piece)-hautbois.ly hautbois.ily
$(piece)-violonUn.pdf: $(piece)-violonUn.ly violonUn.ily
$(piece)-violonDeux.pdf: $(piece)-violonDeux.ly violonDeux.ily

# Lancer `make score' pour générer l'intégrale des quatre mouvements
# en un seul fichier.
.PHONY: score
score: $(piece).pdf
```

```
# Lancer `make parties' pour obtenir tous les pupitres.
# Lancer `make toto.pdf' pour obtenir la partie instrumentale de toto.
# Par exemple : `make symphonie-cello.pdf'.
.PHONY: parties
parties: $(piece)-cello.pdf \
         $(piece)-violonUn.pdf \
         $(piece)-violonDeux.pdf \
         $(piece)-alto.pdf \
         $(piece)-hautbois.pdf \
         $(piece)-cor.pdf

# Lancer `make mouvements' pour générer un fichier séparé pour chacun
# des mouvements.
.PHONY: mouvements
mouvements: $(piece)I.pdf \
            $(piece)II.pdf \
            $(piece)III.pdf \
            $(piece)IV.pdf

all: score parties mouvements

archive:
        tar -cvvf symphonie.tar \    # cette ligne commence par une tabulation
        --exclude=*pdf --exclude=*~ \
        --exclude=*midi --exclude=*.tar \
        ../Symphonie/*
```

Les choses se compliquent sous Windows. Une fois GNU Make pour Windows téléchargé et installé, il vous faudra correctement définir le chemin d'accès au programme *Make* – dans les variables d'environnement du système – afin que l'interpréteur de commandes DOS puisse le localiser. Pour cela, faites un clic droite sur « Poste de travail », choisissez `Propriétés` puis `Avancées`. Cliquez sur `Variables d'environnement` puis, dans l'onglet `Variables système`, mettez `path` en surbrillance et cliquez sur `Modifier`. Ajoutez alors le chemin d'accès complet à l'exécutable de GNU Make, qui devrait ressembler à :

```
C:\Program Files\GnuWin32\bin
```

Il va également falloir adapter le *makefile* aux particularités de l'interpréteur de commandes et à la présence d'espaces dans le nom de certains répertoire de ce système. La cible `archive` est tout bonnement supprimée, puisque Windows ne dispose pas de la commande `tar`. Enfin, les fichiers MIDI ont une extension par défaut propre à Windows.

```
## VERSION POUR WINDOWS
##
piece = symphonie
LILY_CMD = lilypond -ddelete-intermediate-files \
                    -dno-point-and-click \
                    -djob-count=$(NUMBER_OF_PROCESSORS)

#Détermination du nom de CURDIR sous sa forme 8.3
#(solution pour les espaces dans le PATH)
workdir = $(shell for /f "tokens=*" %%b in ("$(CURDIR)") \
          do @echo %%~sb)

.SUFFIXES: .ly .ily .pdf .mid
```

```
VPATH = \
  $(workdir)/Partitions \
  $(workdir)/PDF \
  $(workdir)/Pupitress \
  $(workdir)/Notes

%.pdf %.mid: %.ly
        $(LILY_CMD) $<      # cette ligne commence par une tabulation
        if exist "$*.pdf"  move /Y "$*.pdf"  PDF/ # tabulation au début
        if exist "$*.mid" move /Y "$*.mid" MIDI/  # tabulation au début

notes = \
  cello.ily \
  figures.ily \
  cor.ily \
  hautbois.ily \
  trioCordes.ily \
  alto.ily \
  violonUn.ily \
  violonDeux.ily

$(piece)I.pdf: $(piece)I.ly $(notes)
$(piece)II.pdf: $(piece)II.ly $(notes)
$(piece)III.pdf: $(piece)III.ly $(notes)
$(piece)IV.pdf: $(piece)IV.ly $(notes)

$(piece).pdf: $(piece).ly $(notes)

$(piece)-cello.pdf: $(piece)-cello.ly cello.ily
$(piece)-cor.pdf: $(piece)-cor.ly cor.ily
$(piece)-hautbois.pdf: $(piece)-hautbois.ly hautbois.ily
$(piece)-alto.pdf: $(piece)-alto.ly alto.ily
$(piece)-violonUn.pdf: $(piece)-violonUn.ly violonUn.ily
$(piece)-violonDeux.pdf: $(piece)-violonDeux.ly violonDeux.ily

.PHONY: score
score: $(piece).pdf

.PHONY: parties
parties: $(piece)-cello.pdf \
         $(piece)-violonUn.pdf \
         $(piece)-violonDeux.pdf \
         $(piece)-alto.pdf \
         $(piece)-hautbois.pdf \
         $(piece)-cor.pdf

.PHONY: mouvements
mouvements: $(piece)I.pdf \
           $(piece)II.pdf \
           $(piece)III.pdf \
           $(piece)IV.pdf
```

```
all: score parties mouvements
```

Le *Makefile* suivant convient pour un document `lilypond-book` réalisé avec LaTeX. Ce projet contiendra un index, ce qui nécessitera de lancer une deuxième fois `latex` pour mettre à jour les liens. Les fichiers résultants iront dans le répertoire `out` pour ce qui est des pdf et dans le répertoire `htmlout` pour ce qui est du html.

```
SHELL=/bin/sh
FILE=monprojet
OUTDIR=out
WEBDIR=htmlout
VIEWER=acroread
BROWSER=firefox
LILYBOOK_PDF=lilypond-book --output=$(OUTDIR) --pdf $(FILE).lytex
LILYBOOK_HTML=lilypond-book --output=$(WEBDIR) $(FILE).lytex
PDF=cd $(OUTDIR) && pdflatex $(FILE)
HTML=cd $(WEBDIR) && latex2html $(FILE)
INDEX=cd $(OUTDIR) && makeindex $(FILE)
PREVIEW=$(VIEWER) $(OUTDIR)/$(FILE).pdf &

all: pdf web keep

pdf:
        $(LILYBOOK_PDF)  # tabulation en début de ligne
        $(PDF)           # tabulation en début de ligne
        $(INDEX)         # tabulation en début de ligne
        $(PDF)           # tabulation en début de ligne
        $(PREVIEW)       # tabulation en début de ligne

web:
        $(LILYBOOK_HTML) # tabulation en début de ligne
        $(HTML)          # tabulation en début de ligne
        cp -R $(WEBDIR)/$(FILE)/ ./  # tabulation en début de ligne
        $(BROWSER) $(FILE)/$(FILE).html &  # tabulation en début de ligne

keep: pdf
        cp $(OUTDIR)/$(FILE).pdf $(FILE).pdf  # tabulation en début de ligne

clean:
        rm -rf $(OUTDIR) # tabulation en début de ligne

web-clean:
        rm -rf $(WEBDIR) # tabulation en début de ligne

archive:
        tar -cvvf monprojet.tar \ # tabulation en début de ligne
        --exclude=out/* \
        --exclude=htmlout/* \
        --exclude=monprojet/* \
        --exclude=*midi \
        --exclude=*pdf \
        --exclude=*~ \
```

```
        ../MonProjet/*
```

AVENIR: faire que ça marche sous Windows

Ce *makefile* n'est malheureusement pas opérationnel sous Windows. La seule alternative qui s'offre aux utilisateurs de Windows consiste à créer un fichier de traitement par lot (`.bat`) qui contienne les différentes commandes successives. Bien que cette manière de procéder ne tienne aucun compte des dépendances entre fichiers, elle permet de réduire le nombre de processus à lancer dans une seule commande. Vous devrez enregistrer les lignes suivantes dans un fichier `construire.bat` ou `construire.cmd`. Ce fichier pourra être exécuté soit en ligne de commande, soit par un double clic sur son icône.

```
lilypond-book --output=out --pdf monprojet.lytex
cd out
pdflatex monprojet
makeindex monprojet
pdflatex monprojet
cd ..
copy out\monprojet.pdf MonProjet.pdf
```

Voir aussi

Manuel d'utilisation : Section 1.2 [Utilisation en ligne de commande], page 1, Chapitre 3 [lilypond-book], page 22

Annexe A GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

   The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

   This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

   We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

   This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

   A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

   A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

   The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

   The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

   You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

   You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both

covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its

Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements.”

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `http://www.gnu.org/copyleft/`.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

    "Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

    "CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

    "Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

    An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

    The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with. . . Texts." line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Annexe B Index de LilyPond

R

S

T

U

V

W

X